DMA Controllers

This chapter describes the features and operation of the device System DMA (DMA_SYSTEM) and Enhanced DMA (EDMA) controllers.

NOTE: EVE and VCP are not supported in this family of devices.

Topic **Page**

16.1 System DMA

This chapter describes the system direct memory access (DMA_SYSTEM) module.

NOTE: EVE and VCP are not supported in this family of devices.

16.1.1 DMA_SYSTEM Module Overview

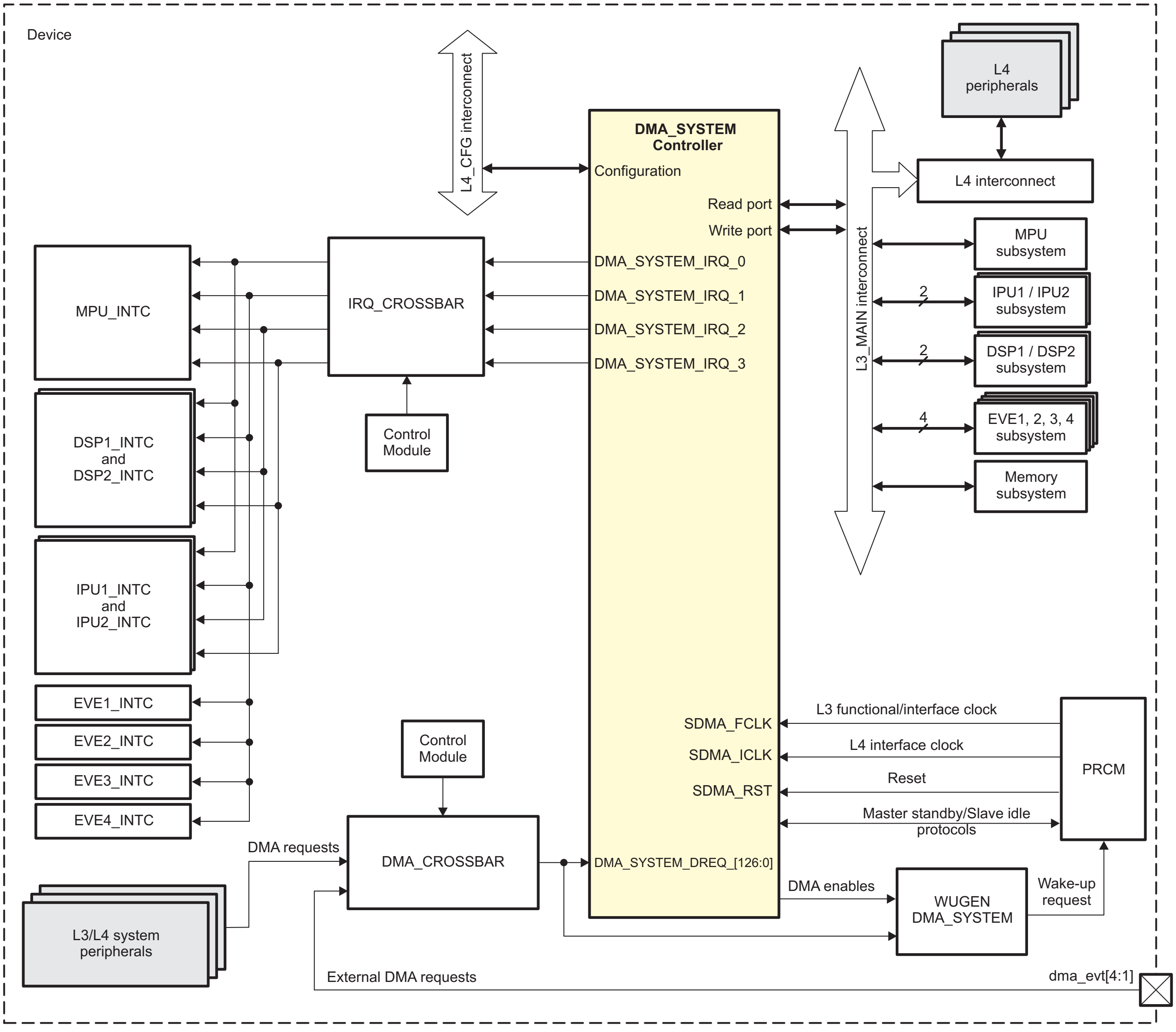
The system direct memory access (DMA_SYSTEM) module, also called DMA4, performs high-performance data transfers between memories and peripheral devices without microprocessor unit (MPU) or digital signal processor (DSP) support during transfer. A DMA transfer is programmed through a logical DMA channel, which allows the transfer to be optimally tailored to the requirements of the application.

The DMA controller includes the following main features:

- Data transfer support in either direction between:
  - Memory and memory
  - Memory and peripheral device
- 32 logical DMA channels supporting:
  - Multiple concurrent transfers
  - Independent transfer profile for each channel
  - 8-bit, 16-bit, or 32-bit data element transfer size
  - Software-triggered or hardware-synchronized transfers
  - Flexible source and destination address generation
  - Burst read and write - max burst size is 16
  - Chained multiple-channel transfers
  - Endianism conversion
  - Draining
  - Linked-list support for descriptor types 1, 2, and 3
- First-come, first-serve DMA scheduling with fixed priority
- Up to 127 Hardware DMA requests
- DMA_CROSSBAR
- Constant fill
- Transparent copy
- Four programmable interrupt request output lines
- FIFO depth: 256 × 64-bit
- Data buffering
- FIFO budget allocation
- Power-management support
- Auto-idle power-saving support

Figure 16-1 shows an overview of the DMA_SYSTEM module.

Figure 16-1. DMA_SYSTEM Overview



The DMA_SYSTEM module has three ports: one read, one write, and one configuration port, and provides multiple logical channel support. A dynamically allocated FIFO queue memory pool provides buffering between the read and write ports, which are multithreaded (two threads for the write port and four threads for the read port); this means that each transaction is flagged by a thread ID (0, 1, 2, or 3) in the request direction and in the response direction. This allows the read port to have four outstanding requests at a time. The write port has two threads budget available.

The MPU (or DSP) configures the DMA_SYSTEM through the L4_CFG interconnect.

16.1.2 DMA_SYSTEM Controller Environment

The DMA_SYSTEM controller supports external DMA requests through the dma_evt[4:1] pins (see Table 16-1). A logical channel can be configured to respond to an external synchronization request.

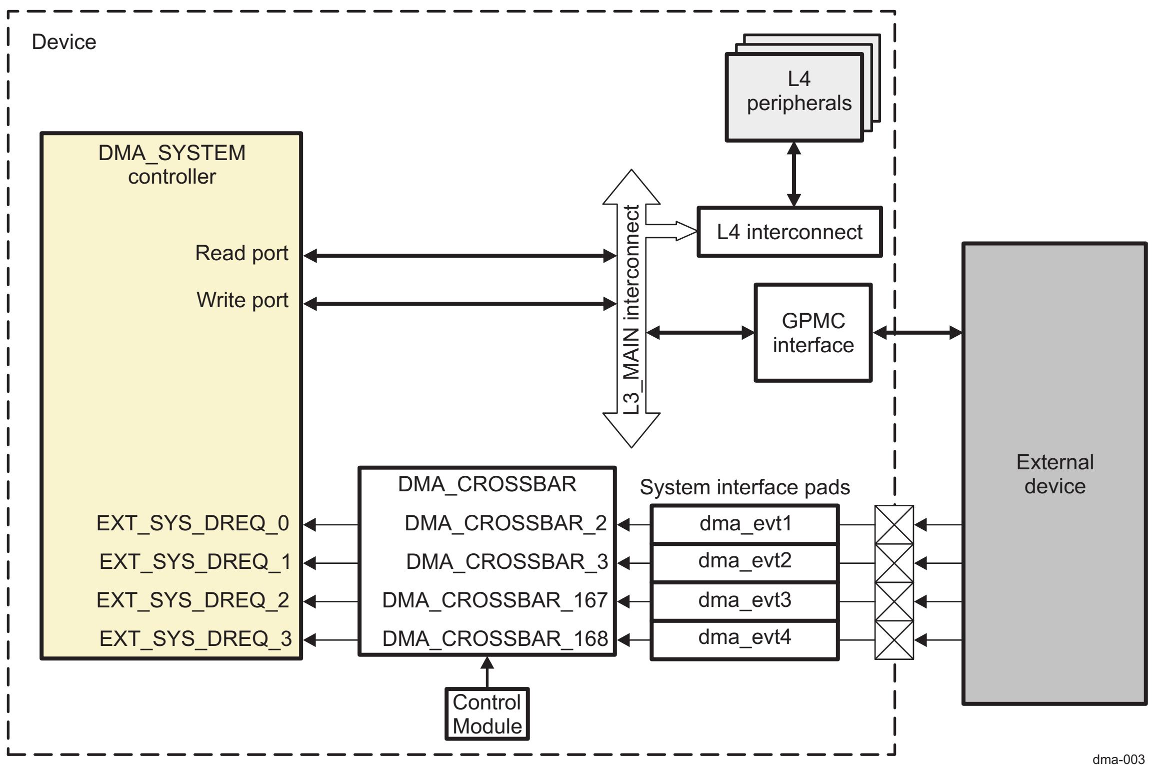
Table 16-1. External DMA_SYSTEM Request Signals

Pin Name	DMA_CROSSBAR Input	Signal Name	I/O [1]	Description	Module Reset Value
dma_evt1	DMA_CROSSBAR_2	EXT_SYS_DREQ_0	I	External DMA request 0 (system expansion)	Z
dma_evt2	DMA_CROSSBAR_3	EXT_SYS_DREQ_1	I	External DMA request 1 (system expansion)	Z
dma_evt3	DMA_CROSSBAR_167	EXT_SYS_DREQ_2	I	External DMA request 2 (system expansion)	Z
dma_evt4	DMA_CROSSBAR_168	EXT_SYS_DREQ_3	I	External DMA request 3 (system expansion)	Z

(1) I = Input, O = Output

Figure 16-2 shows an example of how to use the external hardware DMA request pins in the DMA_SYSTEM environment.

Figure 16-2. Example of External DMA Requests Use



An external device can use the external DMA request pins to start a logical channel transfer over the general-purpose memory controller (GPMC) interface. The transfer can be a memory-to-memory transfer in which the source memory is in the external device.

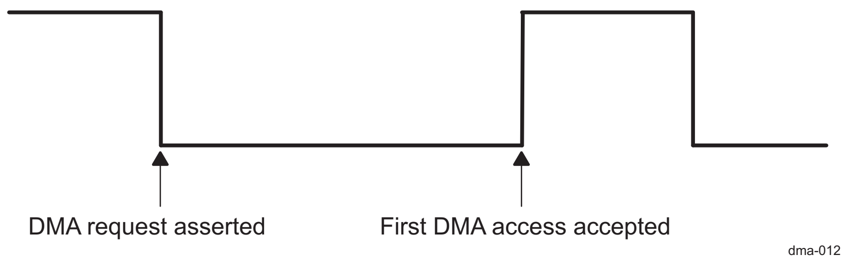
By default, the external DMA request signals are not available on external pins after a cold reset. For more information about multiplexing out the two signal lines to pins, see Section 18.4.6.1.1, *Pad Configuration Registers* in the Chapter 18, *Control Module*.

All 127 DMA request lines are transition sensitive.

For a transition-sensitive DMA request (see Figure 16-3), the line must be maintained low (asserted) until the first DMA access is complete, after which the line must be maintained high (deasserted) for greater than one clock cycle (DMA_L3_GICLK):

- When the deassertion time is less than one clock cycle, the DMA_SYSTEM may not detect the deassertion.
- When the channel is enabled one cycle after a DMA request is disabled, the channel detects the DMA request and starts the corresponding transfer.
- When the channel is enabled two cycles after the DMA request is disabled, the channel does not detect the DMA request.

Figure 16-3. Transition-Sensitive DMA Request Scheme

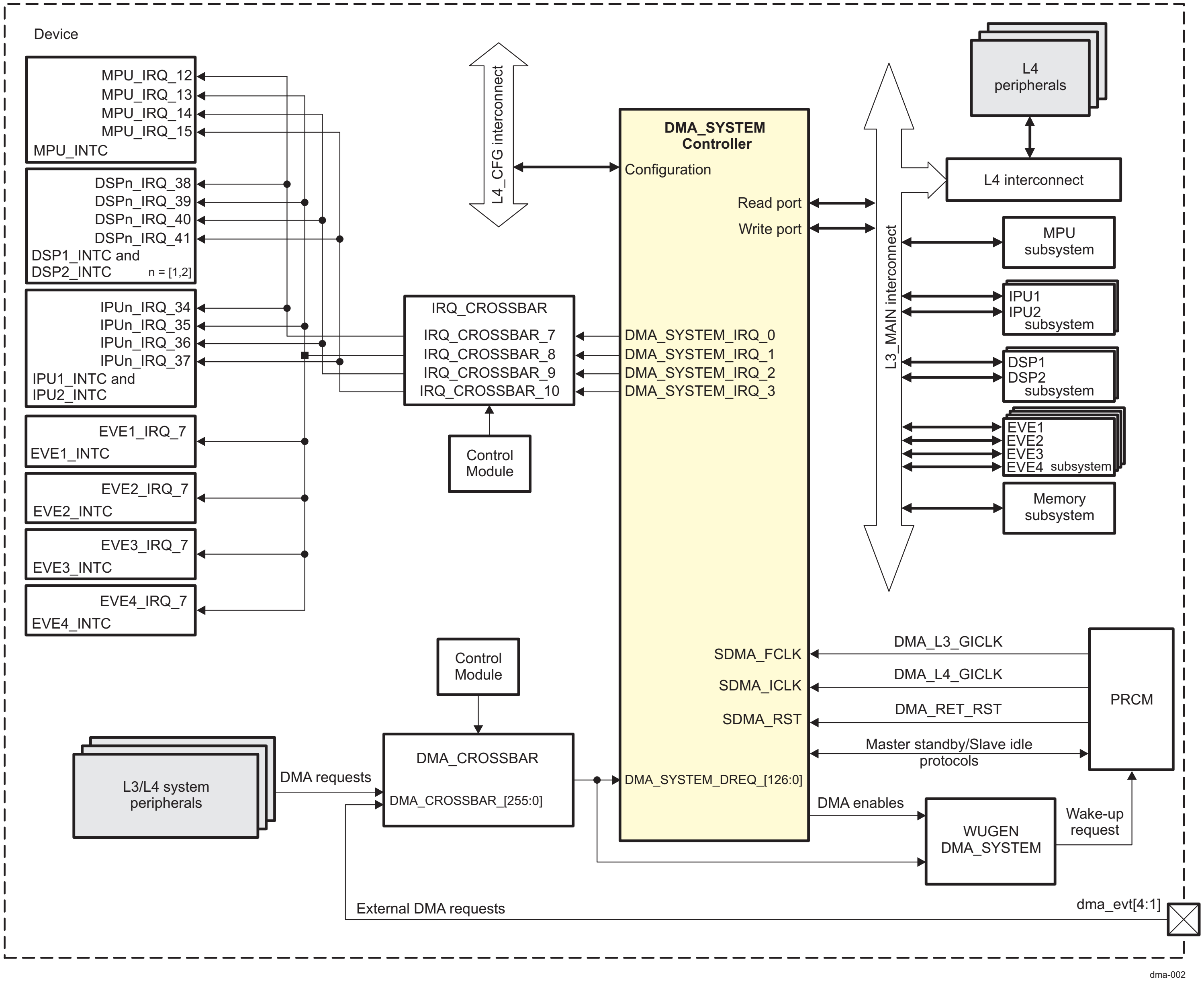


16.1.3 DMA_SYSTEM Module Integration

This section describes the integration of the module in the device, including information about clocks, resets, and hardware requests.

Figure 16-4 shows the DMA_SYSTEM controller integration.

Figure 16-4. DMA_SYSTEM Controller Integration



NOTE: For more information about the system DMA wake-up generator (WUGEN_DMA_SYSTEM), the master standby/slave idle protocols, and the wake-up request, see Section 3.1.1.1, *Clock Management*, in Chapter 3, *Power, Reset, and Clock Management*.

Table 16-2 through Table 16-4 summarize the integration of the module in the device.

Table 16-2. DMA_SYSTEM Integration Attributes

Module Instance	Attributes	
	Power Domain	Interconnect
DMA_SYSTEM	PD_COREAON	L3_MAIN and L4_CFG

Table 16-3. DMA_SYSTEM Clocks and Resets

Clocks				
Module Instance	**Destination Signal Name**	**Source Signal Name**	**Source**	**Description**
DMA_SYSTEM	SDMA_FCLK	DMA_L3_GICLK	PRCM	Functional clock for all internal logic and for the two master read and write ports. For information about the power, reset, and clock management (PRCM) clock gating and management, see Chapter 3, *Power, Reset, and Clock Management*.
	SDMA_ICLK	DMA_L4_GICLK		Interface clock. It supports the configuration port. For information about PRCM clock gating and management, see Chapter 3, *Power, Reset, and Clock Management*.
Resets				
Module Instance	**Destination Signal Name**	**Source Signal Name**	**Source**	**Description**
DMA_SYSTEM	SDMA_RST	DMA_RET_RST	PRCM	Hardware retention reset. It initializes all internal logic of the DMA_SYSTEM module, all global registers, and some of the per-channel registers, implemented in flip-flops. However, all remaining per-channel registers are memory-based, and, therefore, are not reset (have undefined values). Thus, when programming a channel for the first time, all bits that have undefined reset values must be configured before enabling the channel. For information about PRCM reset sources and distribution, see Chapter 3, *Power, Reset, and Clock Management*.

Table 16-4. DMA_SYSTEM Hardware Requests

Interrupt Requests					
Module Instance	**Source Signal Name**	**Destination IRQ_CROSSBAR INPUT**	**Default Mapping**	**Destination**	**Description**

Table 16-4. DMA_SYSTEM Hardware Requests (continued)

DMA_SYSTEM	DMA_SYSTEM_IRQ_0	IRQ_CROSSBAR_7	MPU_IRQ_12	Cortex-A15 MPU INTC	DMA_SYSTEM interrupt request 0. For information about the MPU_INTC, see Section 17.3.1 *Interrupt Requests to MPU_INTC*.
			IPU1_IRQ_34	IPU1 INTC	DMA_SYSTEM interrupt request 0. For information about the IPU1_INTC, see Section 17.3.4*Interrupt Requests to IPU1_Cx_INTC*.
			IPU2_IRQ_34	IPU2 INTC	DMA_SYSTEM interrupt request 0. For information about the IPU2_INTC, see Section 17.3.5 *Interrupt Requests to IPU2_Cx_INTC*.
			DSP1_IRQ_38	DSP1 INTC	DMA_SYSTEM interrupt request 0. For information about the DSP1_INTC, see Section 17.3.2 *Interrupt Requests to DSP1_INTC*.
			DSP2_IRQ_38	DSP2 INTC	DMA_SYSTEM interrupt request 0. For information about the DSP2_INTC, see Section 17.3.3 *Interrupt Requests to DSP2_INTC*.
	DMA_SYSTEM_IRQ_1	IRQ_CROSSBAR_8	MPU_IRQ_13	Cortex-A15 MPU INTC	DMA_SYSTEM interrupt request 1
			IPU1_IRQ_35	IPU1 INTC	
			IPU2_IRQ_35	IPU2 INTC	
			DSP1_IRQ_39	DSP1 INTC	
			DSP2_IRQ_39	DSP2 INTC	
			EVE1_IRQ_7[(1)]	EVE1 INTC	DMA_SYSTEM interrupt request 1. For information about the EVE1_INTC, see Section 17.3.6 *Interrupt Requests to EVE1_INTC*.
			EVE2_IRQ_7[(2)]	EVE2 INTC	DMA_SYSTEM interrupt request 1. For information about the EVE2_INTC, see Section 17.3.7 *Interrupt Requests to EVE2_INTC*.
			EVE3_IRQ_7[(2)]	EVE3 INTC	DMA_SYSTEM interrupt request 1. For information about the EVE3_INTC, see Section 17.3.8 *Interrupt Requests to EVE3_INTC*
			EVE4_IRQ_7[(2)]	EVE4 INTC	DMA_SYSTEM interrupt request 1. For information about the EVE4_INTC, see Section 17.3.9 *Interrupt Requests to EVE4_INTC*
	DMA_SYSTEM_IRQ_2	IRQ_CROSSBAR_9	MPU_IRQ_14	Cortex-A15 MPU INTC	DMA_SYSTEM interrupt request 2

(1) EVE is not supported in this family of devices.
(2) EVE is not supported in this family of devices.

Table 16-4. DMA_SYSTEM Hardware Requests (continued)

DMA_SYSTEM_IRQ_3	IRQ_CROSSBAR_10	IPU1_IRQ_36	IPU1 INTC	DMA_SYSTEM interrupt request 3
		IPU2_IRQ_36	IPU2 INTC	
		DSP1_IRQ_40	DSP1 INTC	
		MPU_IRQ_15	Cortex-A15 MPU INTC	
		IPU1_IRQ_37	IPU1 INTC	
		IPU2_IRQ_37	IPU2 INTC	
		DSP1_IRQ_41	DSP1 INTC	

NOTE: The “**Default Mapping**” column in Table 16-4 *Hardware Requests* shows the default mapping of module IRQ source signals. These IRQ source signals can also be mapped to other lines of each device Interrupt controller through the IRQ_CROSSBAR or DMA_CROSSBAR modules.
For more information about the IRQ_CROSSBAR and DMA_CROSSBAR modules, see sections: Section 18.4.6.4 *IRQ_CROSSBAR Module Functional Description* and *DMA_CROSSBAR Module Functional Description*, in Chapter 18*Control Module*.
For more information about the device interrupt controllers, refer to Chapter 17 *Interrupt Controllers* in the device TRM.

NOTE: For a description of the interrupt source, see Section 16.1.4.2, *DMA_SYSTEM Controller Interrupt Requests*.

16.1.3.1 DMA Requests to the DMA_SYSTEM Controller

Table 16-5 lists the default DMA sources for the DMA_SYSTEM controller. In addition, the DMA_SYSTEM inputs (DMA_SYSTEM_DREQ_[126:0]) can alternatively be sourced through the associated DMA_CROSSBAR from one of the 256 multiplexed device DMA sources listed in Table 16-6. The CTRL_CORE_DMA_SYSTEM_DREQ_y_z registers (where *y* and *z* are indexes of DMA_SYSTEM input lines) in the Control Module are used to select between the default DMA sources and the multiplexed DMA sources.

Table 16-5. DMA_SYSTEM Default Request Mapping

DMA Request Line	DMA_ CROSSBAR Instance Number	DMA_CROSSBAR Configuration Register	DMA_ CROSSBAR Default Input Index	Default DMA Source Name	Default DMA Source Description
DMA_SYSTEM_DREQ_0	1	CTRL_CORE_DMA_SYSTEM_DREQ_0_1[7:0]	1	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_1	2	CTRL_CORE_DMA_SYSTEM_DREQ_0_1[23:16]	2	EXT_SYS_DREQ_0	External DMA request 0 (system expansion)
DMA_SYSTEM_DREQ_2	3	CTRL_CORE_DMA_SYSTEM_DREQ_2_3[7:0]	3	EXT_SYS_DREQ_1	External DMA request 1 (system expansion)
DMA_SYSTEM_DREQ_3	4	CTRL_CORE_DMA_SYSTEM_DREQ_2_3[23:16]	4	GPMC_DREQ	GPMC data transmit request from prefetch engine
DMA_SYSTEM_DREQ_4	5	CTRL_CORE_DMA_SYSTEM_DREQ_4_5[7:0]	5	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_5	6	CTRL_CORE_DMA_SYSTEM_DREQ_4_5[23:16]	6	DISPC_DREQ	Frame update request
DMA_SYSTEM_DREQ_6	7	CTRL_CORE_DMA_SYSTEM_DREQ_6_7[7:0]	7	CT_TBR_DREQ	DEBUG subsystem CT_TBR request
DMA_SYSTEM_DREQ_7	8	CTRL_CORE_DMA_SYSTEM_DREQ_6_7[23:16]	8	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_8	9	CTRL_CORE_DMA_SYSTEM_DREQ_8_9[7:0]	9	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_9	10	CTRL_CORE_DMA_SYSTEM_DREQ_8_9[23:16]	10	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_10	11	CTRL_CORE_DMA_SYSTEM_DREQ_10_11[7:0]	11	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_11	12	CTRL_CORE_DMA_SYSTEM_DREQ_10_11[23:16]	12	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_12	13	CTRL_CORE_DMA_SYSTEM_DREQ_12_13[7:0]	13	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_13	14	CTRL_CORE_DMA_SYSTEM_DREQ_12_13[23:16]	14	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_14	15	CTRL_CORE_DMA_SYSTEM_DREQ_14_15[7:0]	15	MCSPI3_DREQ_TX0	McSPI3 transmit request channel 0
DMA_SYSTEM_DREQ_15	16	CTRL_CORE_DMA_SYSTEM_DREQ_14_15[23:16]	16	MCSPI3_DREQ_RX0	McSPI3 receive request channel 0
DMA_SYSTEM_DREQ_16	17	CTRL_CORE_DMA_SYSTEM_DREQ_16_17[7:0]	17	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_17	18	CTRL_CORE_DMA_SYSTEM_DREQ_16_17[23:16]	18	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_18	19	CTRL_CORE_DMA_SYSTEM_DREQ_18_19[7:0]	19	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_19	20	CTRL_CORE_DMA_SYSTEM_DREQ_18_19[23:16]	20	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_20	21	CTRL_CORE_DMA_SYSTEM_DREQ_20_21[7:0]	21	Reserved	Reserved by default but can be remapped to a valid DMA source

Table 16-5. DMA_SYSTEM Default Request Mapping (continued)

DMA Request Line	DMA_ CROSSBAR Instance Number	DMA_CROSSBAR Configuration Register	DMA_ CROSSBAR Default Input Index	Default DMA Source Name	Default DMA Source Description
DMA_SYSTEM_DREQ_21	22	CTRL_CORE_DMA_SYSTEM_DREQ_20_21[23:16]	22	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_22	23	CTRL_CORE_DMA_SYSTEM_DREQ_22_23[7:0]	23	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_23	24	CTRL_CORE_DMA_SYSTEM_DREQ_22_23[23:16]	24	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_22	23	CTRL_CORE_DMA_SYSTEM_DREQ_22_23[7:0]	23	MCSPI3_DREQ_TX1	McSPI module 3 - transmit request channel 1
DMA_SYSTEM_DREQ_23	24	CTRL_CORE_DMA_SYSTEM_DREQ_22_23[23:16]	24	MCSPI3_DREQ_RX1	McSPI module 3 - receive request channel 1
DMA_SYSTEM_DREQ_24	25	CTRL_CORE_DMA_SYSTEM_DREQ_24_25[7:0]	25	I2C3_DREQ_TX	I2C3 transmit request
DMA_SYSTEM_DREQ_25	26	CTRL_CORE_DMA_SYSTEM_DREQ_24_25[23:16]	26	I2C3_DREQ_RX	I2C3 receive request
DMA_SYSTEM_DREQ_26	27	CTRL_CORE_DMA_SYSTEM_DREQ_26_27[7:0]	27	I2C1_DREQ_TX	I2C1 transmit request
DMA_SYSTEM_DREQ_27	28	CTRL_CORE_DMA_SYSTEM_DREQ_26_27[23:16]	28	I2C1_DREQ_RX	I2C1 receive request
DMA_SYSTEM_DREQ_28	29	CTRL_CORE_DMA_SYSTEM_DREQ_28_29[7:0]	29	I2C2_DREQ_TX	I2C2 transmit request
DMA_SYSTEM_DREQ_29	30	CTRL_CORE_DMA_SYSTEM_DREQ_28_29[23:16]	30	I2C2_DREQ_RX	I2C2 receive request
DMA_SYSTEM_DREQ_30	31	CTRL_CORE_DMA_SYSTEM_DREQ_30_31[7:0]	31	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_31	32	CTRL_CORE_DMA_SYSTEM_DREQ_30_31[23:16]	32	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_32	33	CTRL_CORE_DMA_SYSTEM_DREQ_32_33[7:0]	33	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_33	34	CTRL_CORE_DMA_SYSTEM_DREQ_32_33[23:16]	34	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_34	35	CTRL_CORE_DMA_SYSTEM_DREQ_34_35[7:0]	35	MCSPI1_DREQ_TX0	McSPI1 transmit request channel 0
DMA_SYSTEM_DREQ_35	36	CTRL_CORE_DMA_SYSTEM_DREQ_34_35[23:16]	36	MCSPI1_DREQ_RX0	McSPI1 receive request channel 0
DMA_SYSTEM_DREQ_36	37	CTRL_CORE_DMA_SYSTEM_DREQ_36_37[7:0]	37	MCSPI1_DREQ_TX1	McSPI1 transmit request channel 1
DMA_SYSTEM_DREQ_37	38	CTRL_CORE_DMA_SYSTEM_DREQ_36_37[23:16]	38	MCSPI1_DREQ_RX1	McSPI1 receive request channel 1
DMA_SYSTEM_DREQ_38	39	CTRL_CORE_DMA_SYSTEM_DREQ_38_39[7:0]	39	MCSPI1_DREQ_TX2	McSPI1 transmit request channel 2
DMA_SYSTEM_DREQ_39	40	CTRL_CORE_DMA_SYSTEM_DREQ_38_39[23:16]	40	MCSPI1_DREQ_RX2	McSPI1 receive request channel 2
DMA_SYSTEM_DREQ_40	41	CTRL_CORE_DMA_SYSTEM_DREQ_40_41[7:0]	41	MCSPI1_DREQ_TX3	McSPI1 transmit request channel 3
DMA_SYSTEM_DREQ_41	42	CTRL_CORE_DMA_SYSTEM_DREQ_40_41[23:16]	42	MCSPI1_DREQ_RX3	McSPI1 receive request channel 3
DMA_SYSTEM_DREQ_42	43	CTRL_CORE_DMA_SYSTEM_DREQ_42_43[7:0]	43	MCSPI2_DREQ_TX0	McSPI2 transmit request channel 0
DMA_SYSTEM_DREQ_43	44	CTRL_CORE_DMA_SYSTEM_DREQ_42_43[23:16]	44	MCSPI2_DREQ_RX0	McSPI2 receive request channel 0

Table 16-5. DMA_SYSTEM Default Request Mapping (continued)

DMA Request Line	DMA_ CROSSBAR Instance Number	DMA_CROSSBAR Configuration Register	DMA_ CROSSBAR Default Input Index	Default DMA Source Name	Default DMA Source Description
DMA_SYSTEM_DREQ_44	45	CTRL_CORE_DMA_SYSTEM_DREQ_44_45[7:0]	45	MCSPI2_DREQ_TX1	McSPI2 transmit request channel 1
DMA_SYSTEM_DREQ_45	46	CTRL_CORE_DMA_SYSTEM_DREQ_44_45[23:16]	46	MCSPI2_DREQ_RX1	McSPI2 receive request channel 1
DMA_SYSTEM_DREQ_46	47	CTRL_CORE_DMA_SYSTEM_DREQ_46_47[7:0]	47	MMC2_DREQ_TX	MMC2 transmit request
DMA_SYSTEM_DREQ_47	48	CTRL_CORE_DMA_SYSTEM_DREQ_46_47[23:16]	48	MMC2_DREQ_RX	MMC2 receive request
DMA_SYSTEM_DREQ_48	49	CTRL_CORE_DMA_SYSTEM_DREQ_48_49[7:0]	49	UART1_DREQ_TX	UART1 transmit request
DMA_SYSTEM_DREQ_49	50	CTRL_CORE_DMA_SYSTEM_DREQ_48_49[23:16]	50	UART1_DREQ_RX	UART1 receive request
DMA_SYSTEM_DREQ_50	51	CTRL_CORE_DMA_SYSTEM_DREQ_50_51[7:0]	51	UART2_DREQ_TX	UART2 transmit request
DMA_SYSTEM_DREQ_51	52	CTRL_CORE_DMA_SYSTEM_DREQ_50_51[23:16]	52	UART2_DREQ_RX	UART2 receive request
DMA_SYSTEM_DREQ_52	53	CTRL_CORE_DMA_SYSTEM_DREQ_52_53[7:0]	53	UART3_DREQ_TX	UART3 transmit request
DMA_SYSTEM_DREQ_53	54	CTRL_CORE_DMA_SYSTEM_DREQ_52_53[23:16]	54	UART3_DREQ_RX	UART3 receive request
DMA_SYSTEM_DREQ_54	55	CTRL_CORE_DMA_SYSTEM_DREQ_54_55[7:0]	55	UART4_DREQ_TX	UART4 transmit request
DMA_SYSTEM_DREQ_55	56	CTRL_CORE_DMA_SYSTEM_DREQ_54_55[23:16]	56	UART4_DREQ_RX	UART4 receive request
DMA_SYSTEM_DREQ_56	57	CTRL_CORE_DMA_SYSTEM_DREQ_56_57[7:0]	57	MMC4_DREQ_TX	MMC4 transmit request
DMA_SYSTEM_DREQ_57	58	CTRL_CORE_DMA_SYSTEM_DREQ_56_57[23:16]	58	MMC4_DREQ_RX	MMC4 receive request
DMA_SYSTEM_DREQ_58	59	CTRL_CORE_DMA_SYSTEM_DREQ_58_59[7:0]	59	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_59	60	CTRL_CORE_DMA_SYSTEM_DREQ_58_59[23:16]	60	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_60	61	CTRL_CORE_DMA_SYSTEM_DREQ_60_61[7:0]	61	MMC1_DREQ_TX	MMC1 transmit request
DMA_SYSTEM_DREQ_61	62	CTRL_CORE_DMA_SYSTEM_DREQ_60_61[23:16]	62	MMC1_DREQ_RX	MMC1 receive request
DMA_SYSTEM_DREQ_62	63	CTRL_CORE_DMA_SYSTEM_DREQ_62_63[7:0]	63	UART5_DREQ_TX	UART5 transmit request
DMA_SYSTEM_DREQ_63	64	CTRL_CORE_DMA_SYSTEM_DREQ_62_63[23:16]	64	UART5_DREQ_RX	UART5 receive request
DMA_SYSTEM_DREQ_64	65	CTRL_CORE_DMA_SYSTEM_DREQ_64_65[7:0]	65	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_65	66	CTRL_CORE_DMA_SYSTEM_DREQ_64_65[23:16]	66	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_66	67	CTRL_CORE_DMA_SYSTEM_DREQ_66_67[7:0]	67	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_67	68	CTRL_CORE_DMA_SYSTEM_DREQ_66_67[23:16]	68	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_68	69	CTRL_CORE_DMA_SYSTEM_DREQ_68_69[7:0]	69	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_69	70	CTRL_CORE_DMA_SYSTEM_DREQ_68_69[23:16]	70	MCSPI4_DREQ_TX0	McSPI4 transmit request channel 0
DMA_SYSTEM_DREQ_70	71	CTRL_CORE_DMA_SYSTEM_DREQ_70_71[7:0]	71	MCSPI4_DREQ_RX0	McSPI4 receive request channel 0

Table 16-5. DMA_SYSTEM Default Request Mapping (continued)

DMA Request Line	DMA_ CROSSBAR Instance Number	DMA_CROSSBAR Configuration Register	DMA_ CROSSBAR Default Input Index	Default DMA Source Name	Default DMA Source Description
DMA_SYSTEM_DREQ_71	72	CTRL_CORE_DMA_SYSTEM_DREQ_70_71[23:16]	72	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_72	73	CTRL_CORE_DMA_SYSTEM_DREQ_72_73[7:0]	73	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_73	74	CTRL_CORE_DMA_SYSTEM_DREQ_72_73[23:16]	74	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_74	75	CTRL_CORE_DMA_SYSTEM_DREQ_74_75[7:0]	75	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_75	76	CTRL_CORE_DMA_SYSTEM_DREQ_74_75[23:16]	76	DSS_DREQ	Display subsystem HDMI audio request
DMA_SYSTEM_DREQ_76	77	CTRL_CORE_DMA_SYSTEM_DREQ_76_77[7:0]	77	MMC3_DREQ_TX	MMC3 transmit request
DMA_SYSTEM_DREQ_77	78	CTRL_CORE_DMA_SYSTEM_DREQ_76_77[23:16]	78	MMC3_DREQ_RX	MMC3 receive request
DMA_SYSTEM_DREQ_78	79	CTRL_CORE_DMA_SYSTEM_DREQ_78_79[7:0]	79	UART6_DREQ_TX	UART6 transmit request
DMA_SYSTEM_DREQ_79	80	CTRL_CORE_DMA_SYSTEM_DREQ_78_79[23:16]	80	UART6_DREQ_RX	UART6 receive request
DMA_SYSTEM_DREQ_80	81	CTRL_CORE_DMA_SYSTEM_DREQ_80_81[7:0]	81	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_81	82	CTRL_CORE_DMA_SYSTEM_DREQ_80_81[23:16]	82	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_82	83	CTRL_CORE_DMA_SYSTEM_DREQ_82_83[7:0]	83	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_83	84	CTRL_CORE_DMA_SYSTEM_DREQ_82_83[23:16]	84	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_84	85	CTRL_CORE_DMA_SYSTEM_DREQ_84_85[7:0]	85	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_85	86	CTRL_CORE_DMA_SYSTEM_DREQ_84_85[23:16]	86	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_86	87	CTRL_CORE_DMA_SYSTEM_DREQ_86_87[7:0]	87	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_87	88	CTRL_CORE_DMA_SYSTEM_DREQ_86_87[23:16]	88	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_88	89	CTRL_CORE_DMA_SYSTEM_DREQ_88_89[7:0]	89	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_89	90	CTRL_CORE_DMA_SYSTEM_DREQ_88_89[23:16]	90	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_90	91	CTRL_CORE_DMA_SYSTEM_DREQ_90_91[7:0]	91	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_91	92	CTRL_CORE_DMA_SYSTEM_DREQ_90_91[23:16]	92	Reserved	Reserved by default but can be remapped to a valid DMA source

Table 16-5. DMA_SYSTEM Default Request Mapping (continued)

DMA Request Line	DMA_ CROSSBAR Instance Number	DMA_CROSSBAR Configuration Register	DMA_ CROSSBAR Default Input Index	Default DMA Source Name	Default DMA Source Description
DMA_SYSTEM_DREQ_92	93	CTRL_CORE_DMA_SYSTEM_DREQ_92_93[7:0]	93	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_93	94	CTRL_CORE_DMA_SYSTEM_DREQ_92_93[23:16]	94	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_94	95	CTRL_CORE_DMA_SYSTEM_DREQ_94_95[7:0]	95	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_95	96	CTRL_CORE_DMA_SYSTEM_DREQ_94_95[23:16]	96	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_96	97	CTRL_CORE_DMA_SYSTEM_DREQ_96_97[7:0]	97	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_97	98	CTRL_CORE_DMA_SYSTEM_DREQ_96_97[23:16]	98	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_98	99	CTRL_CORE_DMA_SYSTEM_DREQ_98_99[7:0]	99	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_99	100	CTRL_CORE_DMA_SYSTEM_DREQ_98_99[23:16]	100	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_100	101	CTRL_CORE_DMA_SYSTEM_DREQ_100_101[7:0]	101	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_101	102	CTRL_CORE_DMA_SYSTEM_DREQ_100_101[23:16]	102	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_102	103	CTRL_CORE_DMA_SYSTEM_DREQ_102_103[7:0]	103	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_103	104	CTRL_CORE_DMA_SYSTEM_DREQ_102_103[23:16]	104	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_104	105	CTRL_CORE_DMA_SYSTEM_DREQ_104_105[7:0]	105	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_105	106	CTRL_CORE_DMA_SYSTEM_DREQ_104_105[23:16]	106	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_106	107	CTRL_CORE_DMA_SYSTEM_DREQ_106_107[7:0]	107	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_107	108	CTRL_CORE_DMA_SYSTEM_DREQ_106_107[23:16]	108	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_108	109	CTRL_CORE_DMA_SYSTEM_DREQ_108_109[7:0]	109	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_109	110	CTRL_CORE_DMA_SYSTEM_DREQ_108_109[23:16]	110	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_110	111	CTRL_CORE_DMA_SYSTEM_DREQ_110_111[7:0]	111	Reserved	Reserved by default but can be remapped to a valid DMA source

Table 16-5. DMA_SYSTEM Default Request Mapping (continued)

DMA Request Line	DMA_ CROSSBAR Instance Number	DMA_CROSSBAR Configuration Register	DMA_ CROSSBAR Default Input Index	Default DMA Source Name	Default DMA Source Description
DMA_SYSTEM_DREQ_111	112	CTRL_CORE_DMA_SYSTEM_DREQ_110_111[23:16]	112	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_112	113	CTRL_CORE_DMA_SYSTEM_DREQ_112_113[7:0]	113	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_113	114	CTRL_CORE_DMA_SYSTEM_DREQ_112_113[23:16]	114	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_114	115	CTRL_CORE_DMA_SYSTEM_DREQ_114_115[7:0]	115	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_115	116	CTRL_CORE_DMA_SYSTEM_DREQ_114_115[23:16]	116	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_116	117	CTRL_CORE_DMA_SYSTEM_DREQ_116_117[7:0]	117	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_117	118	CTRL_CORE_DMA_SYSTEM_DREQ_116_117[23:16]	118	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_118	119	CTRL_CORE_DMA_SYSTEM_DREQ_118_119[7:0]	119	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_119	120	CTRL_CORE_DMA_SYSTEM_DREQ_118_119[23:16]	120	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_120	121	CTRL_CORE_DMA_SYSTEM_DREQ_120_121[7:0]	121	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_121	122	CTRL_CORE_DMA_SYSTEM_DREQ_120_121[23:16]	122	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_122	123	CTRL_CORE_DMA_SYSTEM_DREQ_122_123[7:0]	123	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_123	124	CTRL_CORE_DMA_SYSTEM_DREQ_122_123[23:16]	124	I2C4_DREQ_TX	I2C4 transmit request
DMA_SYSTEM_DREQ_124	125	CTRL_CORE_DMA_SYSTEM_DREQ_124_125[7:0]	125	I2C4_DREQ_RX	I2C4 receive request
DMA_SYSTEM_DREQ_125	126	CTRL_CORE_DMA_SYSTEM_DREQ_124_125[23:16]	126	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_SYSTEM_DREQ_126	127	CTRL_CORE_DMA_SYSTEM_DREQ_126_127[7:0]	127	Reserved	Reserved by default but can be remapped to a valid DMA source

16.1.3.2 Mapping of DMA Requests to DMA_CROSSBAR Inputs

NOTE: For information about the DMA_CROSSBAR module, refer to Section 18.4.6.5, *DMA_CROSSBAR Module Functional Description* in Chapter 18, *Control Module*.

Table 16-6 shows the mapping of device DMA requests to DMA_CROSSBAR inputs.

Table 16-6. Connection of The Device DREQs to The DMA_CROSSBAR Inputs

DMA_CROSSBAR Input	Device Module DREQs	Description
DMA_CROSSBAR_0	Reserved	Reserved
DMA_CROSSBAR_1	Reserved	Reserved
DMA_CROSSBAR_2	EXT_SYS_DREQ_0	External DMA request 0 (system expansion) - coming from SOC IOs. Level sensitive only
DMA_CROSSBAR_3	EXT_SYS_DREQ_1	External DMA request 1 (system expansion) - coming from SOC IOs - level sentitive only
DMA_CROSSBAR_4	GPMC_DREQ	GPMC request from prefetch engine
DMA_CROSSBAR_5	Reserved	Reserved
DMA_CROSSBAR_6	DISPC_DREQ	The line trigger signal to synchronize a memory to memory logical channel in the DMA4 (system DMA) is generated by the Display Controller IP.
DMA_CROSSBAR_7	CT_TBR_DREQ	DMA request coming from CT_TBR in DEBUGSS (used to be External DMA request 2 - coming from SOC IOs)
DMA_CROSSBAR_8 to DMA_CROSSBAR_14	Reserved	Reserved
DMA_CROSSBAR_15	MCSPI3_DREQ_TX0	McSPI module 3 - transmit request channel 0
DMA_CROSSBAR_16	MCSPI3_DREQ_RX0	McSPI module 3 - receive request channel 0
DMA_CROSSBAR_17 to DMA_CROSSBAR_22	Reserved	Reserved
DMA_CROSSBAR_23	MCSPI3_DREQ_TX1	McSPI module 3 - transmit request channel 1
DMA_CROSSBAR_24	MCSPI3_DREQ_RX1	McSPI module 3 - receive request channel 1
DMA_CROSSBAR_25	I2C3_DREQ_TX	I2C module 3 - transmit request
DMA_CROSSBAR_26	I2C3_DREQ_RX	I2C module 3 - receive request
DMA_CROSSBAR_27	I2C1_DREQ_TX	I2C module 1 - transmit request
DMA_CROSSBAR_28	I2C1_DREQ_RX	I2C module 1 - receive request
DMA_CROSSBAR_29	I2C2_DREQ_TX	I2C module 2 - transmit request
DMA_CROSSBAR_30	I2C2_DREQ_RX	I2C module 2 - receive request
DMA_CROSSBAR_31 to DMA_CROSSBAR_34	Reserved	Reserved
DMA_CROSSBAR_35	MCSPI1_DREQ_TX0	McSPI module 1 - transmit request channel 0
DMA_CROSSBAR_36	MCSPI1_DREQ_RX0	McSPI module 1 - receive request channel 0
DMA_CROSSBAR_37	MCSPI1_DREQ_TX1	McSPI module 1 - transmit request channel 1
DMA_CROSSBAR_38	MCSPI1_DREQ_RX1	McSPI module 1 - receive request channel 1
DMA_CROSSBAR_39	MCSPI1_DREQ_TX2	McSPI module 1 - transmit request channel 2
DMA_CROSSBAR_40	MCSPI1_DREQ_RX2	McSPI module 1 - receive request channel 2
DMA_CROSSBAR_41	MCSPI1_DREQ_TX3	McSPI module 1 - transmit request channel 3
DMA_CROSSBAR_42	MCSPI1_DREQ_RX3	McSPI module 1 - receive request channel 3
DMA_CROSSBAR_43	MCSPI2_DREQ_TX0	McSPI module 2 - transmit request channel 0
DMA_CROSSBAR_44	MCSPI2_DREQ_RX0	McSPI module 2 - receive request channel 0
DMA_CROSSBAR_45	MCSPI2_DREQ_TX1	McSPI module 2 - transmit request channel 1
DMA_CROSSBAR_46	MCSPI2_DREQ_RX1	McSPI module 2 - receive request channel 1
DMA_CROSSBAR_47	MMC2_DREQ_TX	MMC/SD2 transmit request
DMA_CROSSBAR_48	MMC2_DREQ_RX	MMC/SD2 receive request
DMA_CROSSBAR_49	UART1_DREQ_TX	UART module 1 - transmit request

Table 16-6. Connection of The Device DREQs to The DMA_CROSSBAR Inputs (continued)

DMA_CROSSBAR Input	Device Module DREQs	Description
DMA_CROSSBAR_50	UART1_DREQ_RX	UART module 1 - receive request
DMA_CROSSBAR_51	UART2_DREQ_TX	UART module 2 - transmit request
DMA_CROSSBAR_52	UART2_DREQ_RX	UART module 2 - receive request
DMA_CROSSBAR_53	UART3_DREQ_TX	UART module 3 - transmit request (Also infrared - IRDA)
DMA_CROSSBAR_54	UART3_DREQ_RX	UART module 3 - receive request (Also infrared - IRDA)
DMA_CROSSBAR_55	UART4_DREQ_TX	UART module 4 – transmit request
DMA_CROSSBAR_56	UART4_DREQ_RX	UART module 4 – receive request
DMA_CROSSBAR_57	MMC4_DREQ_TX	MMC/SD4 transmit request
DMA_CROSSBAR_58	MMC4_DREQ_RX	MMC/SD4 receive request
DMA_CROSSBAR_59	Reserved	Reserved
DMA_CROSSBAR_60	Reserved	Reserved
DMA_CROSSBAR_61	MMC1_DREQ_TX	MMC/SD1 transmit request
DMA_CROSSBAR_62	MMC1_DREQ_RX	MMC/SD1 receive request
DMA_CROSSBAR_63	UART5_DREQ_TX	UART module 5 – transmit request (used to be External DMA request 3 - coming from SOC IOs)
DMA_CROSSBAR_64	UART5_DREQ_RX	UART module 5 – receive request (used to be USIM receive request)
DMA_CROSSBAR_65 to DMA_CROSSBAR_69	Reserved	Reserved
DMA_CROSSBAR_70	MCSPI4_DREQ_TX0	McSPI module 4 - transmit request channel 0
DMA_CROSSBAR_71	MCSPI4_DREQ_RX0	McSPI module 4 - receive request channel 0
DMA_CROSSBAR_72 to DMA_CROSSBAR_75	Reserved	Reserved
DMA_CROSSBAR_76	DSS_DREQ	Display subsystem HDMI Audio DMA request
DMA_CROSSBAR_77	MMC3_DREQ_TX	MMC/SD3 transmit request
DMA_CROSSBAR_78	MMC3_DREQ_RX	MMC/SD3 receive request
DMA_CROSSBAR_79	UART6_DREQ_TX	UART module 6 – transmit request (used to be USIM transmit request)
DMA_CROSSBAR_80	UART6_DREQ_RX	UART module 6 – receive request (used to be USIM receive request)
DMA_CROSSBAR_81 to DMA_CROSSBAR_123	Reserved	Reserved
DMA_CROSSBAR_124	I2C4_DREQ_TX	I2C module 4 - transmit request
DMA_CROSSBAR_125	I2C4_DREQ_RX	I2C module 4 - receive request
DMA_CROSSBAR_126	Reserved	Reserved
DMA_CROSSBAR_127	Reserved	Reserved
DMA_CROSSBAR_128	McASP1_DREQ_RX	McASP receive event
DMA_CROSSBAR_129	McASP1_DREQ_TX	McASP transmit event
DMA_CROSSBAR_130	McASP2_DREQ_RX	McASP receive event
DMA_CROSSBAR_131	McASP2_DREQ_TX	McASP transmit event
DMA_CROSSBAR_132	McASP3_DREQ_RX	McASP receive event
DMA_CROSSBAR_133	McASP3_DREQ_TX	McASP transmit event
DMA_CROSSBAR_134	McASP4_DREQ_RX	McASP receive event
DMA_CROSSBAR_135	McASP4_DREQ_TX	McASP transmit event
DMA_CROSSBAR_136	McASP5_DREQ_RX	McASP receive event
DMA_CROSSBAR_137	McASP5_DREQ_TX	McASP transmit event
DMA_CROSSBAR_138	McASP6_DREQ_RX	McASP receive event
DMA_CROSSBAR_139	McASP6_DREQ_TX	McASP transmit event
DMA_CROSSBAR_140	McASP7_DREQ_RX	McASP receive event

Table 16-6. Connection of The Device DREQs to The DMA_CROSSBAR Inputs (continued)

DMA_CROSSBAR Input	Device Module DREQs	Description
DMA_CROSSBAR_141	McASP7_DREQ_TX	McASP transmit event
DMA_CROSSBAR_142	McASP8_DREQ_RX	McASP receive event
DMA_CROSSBAR_143	McASP8_DREQ_TX	McASP receive event
DMA_CROSSBAR_144	UART7_DREQ_TX	UART module 7 - transmit request
DMA_CROSSBAR_145	UART7_DREQ_RX	UART module 7 - receive request
DMA_CROSSBAR_146	UART8_DREQ_TX	UART module 8 - transmit request
DMA_CROSSBAR_147	UART8_DREQ_RX	UART module 8 - receive request
DMA_CROSSBAR_148	UART9_DREQ_TX	UART module 9 - transmit request
DMA_CROSSBAR_149	UART9_DREQ_RX	UART module 9 - receive request
DMA_CROSSBAR_150	UART10_DREQ_TX	UART module 10 - transmit request
DMA_CROSSBAR_151	UART10_DREQ_RX	UART module 10 - receive request
DMA_CROSSBAR_152	I2C5_DREQ_TX	I2C module 5 - transmit request
DMA_CROSSBAR_153	I2C5_DREQ_RX	I2C module 5 - receive request
DMA_CROSSBAR_154	VCP1_DREQ_RX(1)	VCP RX Event
DMA_CROSSBAR_155	VCP1_DREQ_TX(1)	VCP TX Event
DMA_CROSSBAR_156	VCP2_DREQ_RX(1)	VCP RX Event
DMA_CROSSBAR_157	VCP2_DREQ_TX(1)	VCP TX Event
DMA_CROSSBAR_158	DCAN1_DREQ_IF1	DCAN IF1 Event
DMA_CROSSBAR_159	DCAN1_DREQ_IF2	DCAN IF2 Event
DMA_CROSSBAR_160	DCAN1_DREQ_IF3	DCAN IF3 Event
DMA_CROSSBAR_161	DCAN2_DREQ_IF1	DCAN IF1 Event
DMA_CROSSBAR_162	DCAN2_DREQ_IF2	DCAN IF2 Event
DMA_CROSSBAR_163	DCAN2_DREQ_IF3	DCAN IF3 Event
DMA_CROSSBAR_164 to DMA_CROSSBAR_166	Reserved	Reserved
DMA_CROSSBAR_167	EXT_SYS_DREQ_2	External DMA request 2 (system expansion) - coming from SOC IOs. Level sensitive only
DMA_CROSSBAR_168	EXT_SYS_DREQ_3	External DMA request 3 (system expansion) - coming from SOC IOs. Level sentitive only
DMA_CROSSBAR_169	MCSPI2_DREQ_TX2	McSPI module 2 - transmit request channel 2
DMA_CROSSBAR_170	MCSPI2_DREQ_RX2	McSPI module 2 - receive request channel 2
DMA_CROSSBAR_171	MCSPI2_DREQ_TX3	McSPI module 2 - transmit request channel 3
DMA_CROSSBAR_172	MCSPI2_DREQ_RX3	McSPI module 2 - receive request channel 3
DMA_CROSSBAR_173	MCSPI3_DREQ_TX2	McSPI module 3 - transmit request channel 2
DMA_CROSSBAR_174	MCSPI3_DREQ_RX2	McSPI module 3 - receive request channel 2
DMA_CROSSBAR_175	MCSPI3_DREQ_TX3	McSPI module 3 - transmit request channel 3
DMA_CROSSBAR_176	MCSPI3_DREQ_RX3	McSPI module 3 - receive request channel 3
DMA_CROSSBAR_177	MCSPI4_DREQ_TX1	McSPI module 4 - transmit request channel 1
DMA_CROSSBAR_178	MCSPI4_DREQ_RX1	McSPI module 4 - receive request channel 1
DMA_CROSSBAR_179	MCSPI4_DREQ_TX2	McSPI module 4 - transmit request channel 2
DMA_CROSSBAR_180	MCSPI4_DREQ_RX2	McSPI module 4 - receive request channel 2
DMA_CROSSBAR_181	MCSPI4_DREQ_TX3	McSPI module 4 - transmit request channel 3
DMA_CROSSBAR_182	MCSPI4_DREQ_RX3	McSPI module 4 - receive request channel 3
DMA_CROSSBAR_183	PRUSS1_DREQ_HOST_REQ0	PRU-ICSS1 Host DMA request 0 (mapped to PRU-ICSS1 Host Interrupt 9)
DMA_CROSSBAR_184	PRUSS1_DREQ_HOST_REQ1	PRU-ICSS1 Host DMA request 1 (mapped to PRU-ICSS1 Host Interrupt 8)

(1) VCP1 and VCP2 are not supported in this family of devices.

Table 16-6. Connection of The Device DREQs to The DMA_CROSSBAR Inputs (continued)

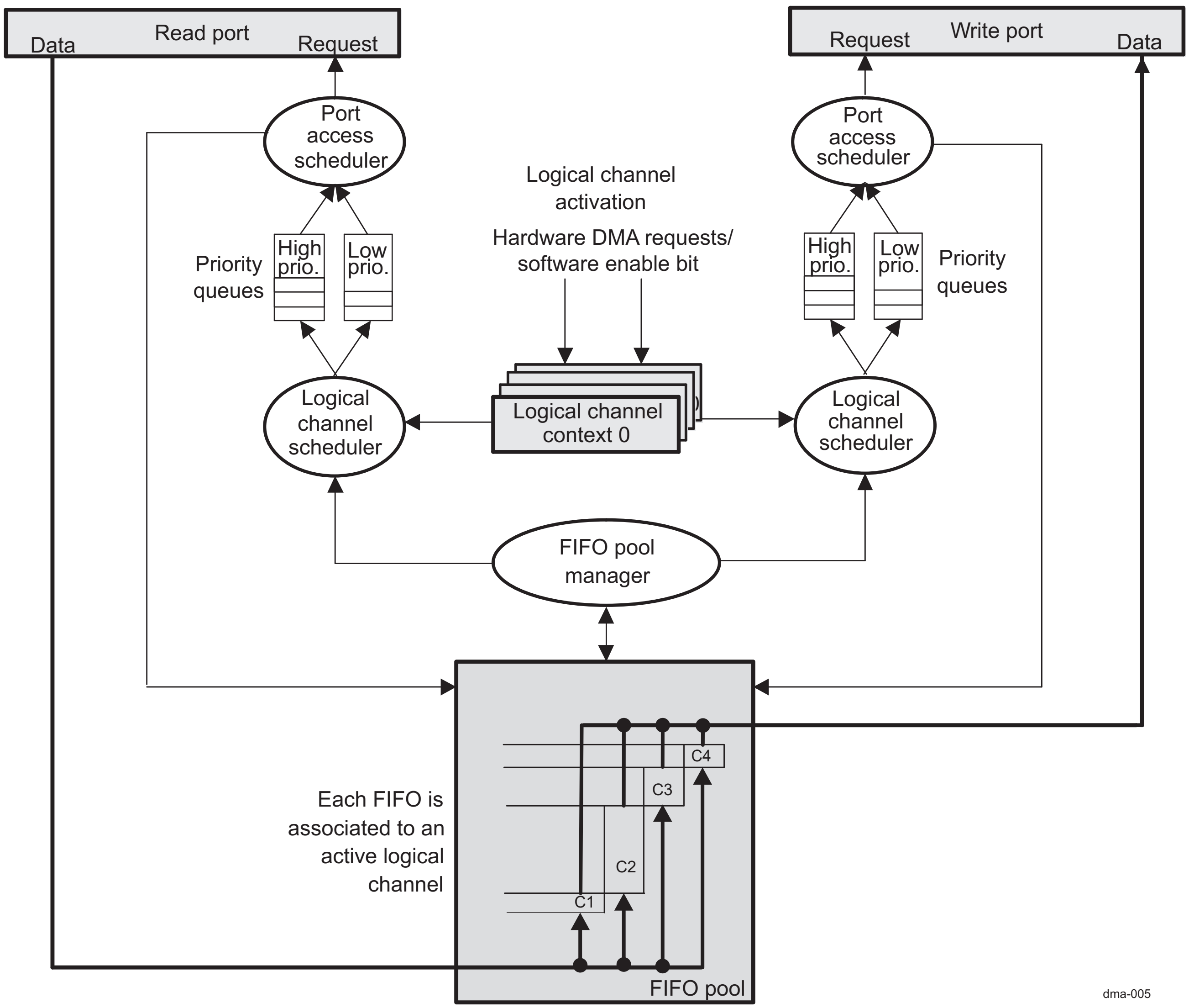
DMA_CROSSBAR Input	Device Module DREQs	Description
DMA_CROSSBAR_185	PRUSS2_DREQ_HOST_REQ0	PRU-ICSS2 Host DMA request 0 (mapped to PRU-ICSS2 Host Interrupt 9)
DMA_CROSSBAR_186	PRUSS2_DREQ_HOST_REQ1	PRU-ICSS2 Host DMA request 1 (mapped to PRU-ICSS2 Host Interrupt 8)
DMA_CROSSBAR_187	GPIO1_DREQ_EVT	GPIO module 1 - event/interrupt 1
DMA_CROSSBAR_188	GPIO2_DREQ_EVT	GPIO module 2 - event/interrupt 1
DMA_CROSSBAR_189	GPIO3_DREQ_EVT	GPIO module 3 - event/interrupt 1
DMA_CROSSBAR_190	GPIO4_DREQ_EVT	GPIO module 4 - event/interrupt 1
DMA_CROSSBAR_191	GPIO5_DREQ_EVT	GPIO module 5 - event/interrupt 1
DMA_CROSSBAR_192	GPIO6_DREQ_EVT	GPIO module 6 - event/interrupt 1
DMA_CROSSBAR_193	GPIO7_DREQ_EVT	GPIO module 7 - event/interrupt 1
DMA_CROSSBAR_194	GPIO8_DREQ_EVT	GPIO module 8 - event/interrupt 1
DMA_CROSSBAR_195	PWMSS1_DREQ_ePWM0_EVT	eHRPWM0 event/interrupt
DMA_CROSSBAR_196	PWMSS2_DREQ_ePWM1_EVT	eHRPWM1 event/interrupt
DMA_CROSSBAR_197	PWMSS3_DREQ_ePWM2_EVT	eHRPWM2 event/interrupt
DMA_CROSSBAR_198	PWMSS1_DREQ_eQEP0_EVT	eQEP0 event/interrupt
DMA_CROSSBAR_199	PWMSS2_DREQ_eQEP1_EVT	eQEP1 event/interrupt
DMA_CROSSBAR_200	PWMSS3_DREQ_eQEP2_EVT	eQEP2 event/interrupt
DMA_CROSSBAR_201	PWMSS1_DREQ_eCAP0_EVT	eCAP0 event/interrupt
DMA_CROSSBAR_202	PWMSS2_DREQ_eCAP1_EVT	eCAP1 event/interrupt
DMA_CROSSBAR_203	PWMSS3_DREQ_eCAP2_EVT	eCAP2 event/interrupt
DMA_CROSSBAR_204 to DMA_CROSSBAR_255	Reserved	Reserved

16.1.4 DMA_SYSTEM Functional Description

The DMA_SYSTEM module provides high-performance data transfers between memories and peripheral devices with low processor use. A DMA transfer is programmed through a logical DMA channel, which allows the transfer to be optimally tailored to the requirements of the application.

Figure 16-5 shows the DMA_SYSTEM controller top-level block diagram.

Figure 16-5. DMA_SYSTEM Controller Top-Level Block Diagram



16.1.4.1 DMA_SYSTEM Controller Power Management

Table 16-7 describes power-management features available for the DMA_SYSTEM controller.

NOTE:

- For information about source clock gating and sleep/wake-up transitions, see secton Section 3.1.1.1 *Clock Management*, in Chapter 3, *Power, Reset, and Clock Management*.
- For a description of the EnaWakeUp, IdleMode, ClockActivity, and StandbyMode features, see Section 3.1.1.1.2 *Module Level Clock Management*, in Chapter 3, *Power, Reset, and Clock Management*.

Table 16-7. Local Power-Management Features

Feature	Registers	Description
Clock auto gating	DMA4_OCP_SYSCONFIG[0] AUTOIDLE bit	This bit allows local power optimization inside the module by gating the SDMA_ICLK clock upon the interface activity.
Slave idle modes	DMA4_OCP_SYSCONFIG[4:3] SIDLEMODE bit field	Force-idle, no-idle, and smart-idle modes are available.
Clock activity	DMA4_OCP_SYSCONFIG[9:8] CLOCKACTIVITY bit field	For configuration details, see Table 16-8.
Master standby modes	DMA4_OCP_SYSCONFIG[13:12] MIDLEMODE bit field	Force-standby, no-standby, and smart-standby modes are available.
Global wake-up enable	N/A	Feature not available
Wake-up sources enable	N/A	Feature not available

Table 16-8. Clock Activity Settings

SDMA_CLOCKACTIVITY Values	Clock State When Module is in IDLE State	
	SDMA_ICLK	SDMA_FCLK
00	Off	Off
10	Off	On
01	On	Off
11	On	On

CAUTION

Because the PRCM module cannot read CLOCKACTIVITY settings through hardware, software must ensure consistent programming between the SDMA_CLOCKACTIVITY and DMA_SYSTEM clock PRCM control bits. For a description of the ClockActivity feature, see Section 3.1.1.1.2, *Module Level Clock Management*, in Chapter 3, *Power, Reset, and Clock Management*.

16.1.4.2 DMA_SYSTEM Controller Interrupt Requests

DMA4 has four interrupt lines (Lj, where j = 0, 1, 2, 3). Each logical channel can request an interrupt over any line. The attachment of a channel interrupt event to one of these four external lines is programmable. Software determines whether it attaches a channel interrupt to a single IRQ line or to multiple IRQ lines.

There are two different registers per interrupt line:

- The DMA4_IRQSTATUS_Lj CH_31_0_Lj field shows the status of the different sources of interrupt. If the DMA4_IRQENABLE_Lj bit is 1, channel *i* is the source of interrupt in line *j*. In contrast to the DMA4_CSRi registers, the DMA4_IRQSTATUS_Lj registers are updated regardless of the corresponding bits in the DMA4_IRQENABLE_Lj registers.
- The DMA4_IRQENABLE_Lj CH_31_0_Lj_EN field masks/unmasks the channel interrupt. If the DMA4_IRQENABLE_Lj bit is set to 0, channel interrupt *i* of the line *j* is masked.

Each logical channel can generate a number of different interrupt events when enabled (that is, set to 1) in the DMA4_CICRi register. Each status bit is updated in the DMA4_CSRi register only when the corresponding enable bit is enabled in the DMA4_CICRi register.

To determine an interrupt source when an interrupt rises on an interrupt line L*j*:

- Identify the channel (LCH*i*) generating the interrupt.

  Read DMA4_IRQSTATUS_Lj.LCH*i* (LCH0 to LCH31). If LCH*i* = 1, channel *i* is the originator of the interrupt.
- Identify the interrupt event.

Read the LCH*i* DMA4_CSRi. For example, if the drop event (the DMA4_CSRi[1] DROP bit) is 1, a request collision will occur.

The interrupt event status bit in the DMA4_CSRi register is immediately reset after it is written to 1.

The interrupt status bit in the DMA4_IRQSTATUS_Lj register is cleared after it is written to 1.

16.1.4.2.1 Interrupt Generation

The DMA_SYSTEM module has four interrupt request output lines, DMA_SYSTEM_IRQ_0 to DMA_SYSTEM_IRQ_3. One or more logical channels can be programmed to generate an interrupt request on any of these lines when any one of the maskable DMA events listed in Table 16-9 occurs.

Table 16-9. Logical DMA Channel Events

Event	Description
End of packet	A packet transfer completed.
End of block	A block transfer completed.
End of frame	A frame transfer completed.
End of super block	A super block transfer completed.
Half of frame	Half of the current frame transferred.
Start of last frame	The first element of the last frame transferred.
Transaction error	A transaction error is returned by the interconnect in either the read or write port.
Address error	An attempt was made to perform a DMA access to an address not aligned on an ES boundary. Condition to occur: if DMA4_CENi[23:0] CHANNEL_ELMNT_NBR = 0x000000 or DMA4_CFNi[15:0] CHANNEL_FRAME_NBR = 0x0000 or DMA4_CSDPi[1:0] DATA_TYPE = 0x3.
Supervisor transaction error	An error occurred, for example, when an unauthorized initiator (that is not a supervisor) tries to use a supervisor transfer.
Drain end	Drain is completed (DMA4_CCRi[10] WR_ACTIVE becomes 0).
Drop error	A drop event interrupt is generated when a DMA request is being serviced while a second one is asserted and a third one arrives before the second DMA request is serviced.

The logical DMA channels that generate an interrupt on a particular IRQ output are specified through the DMA4_IRQENABLE_Lj register (where *j* is the IRQ number: 0, 1, 2, or 3). The events that generate an interrupt for a particular channel can be configured through the channel DMA4_CICRi register.

When an interrupt is detected, the logical DMA channel generating the event can first be identified by reading the DMA4_IRQSTATUS_Lj register. The event causing the interrupt then can be identified by reading the interrupt status via the relevant DMA channel DMA4_CSRi register.

16.1.4.3 Logical Channel Transfer Overview

As Figure 16-5 shows, the DMA_SYSTEM module has one read port and one write port operating independently of one another. Buffering is provided between the read and write ports through a FIFO queue memory pool that is shared dynamically between the active logical channels.

- Logical channel synchronization

  A logical channel is described as hardware-synchronized when the DMA transfers are triggered by DMA requests from a hardware device. Alternatively, a logical channel is described as nonsynchronized when the DMA transfer is triggered by software.

- Logical channel activation

  A logical channel becomes active as follows:

  – For hardware-synchronized transfers, when the logical channel is enabled and the hardware DMA request line is asserted

  – For software-triggered (nonsynchronized) transfers, as soon as software enables the logical channel

- Logical channel transfer composition

  A DMA transfer is divided automatically into a number of transactions. Depending on the logical channel context configured, the transfer size, start address alignment, addressing mode, and configured maximum burst size, each transaction can be a single access or a burst of accesses.

- Logical channel scheduling

  When several logical channels are active at the same time, schedulers manage the read and write ports. The scheduling of logical channel transfers is similar for both read and write ports. When a logical channel becomes active, it is added to the tail of a scheduling queue. If more than one logical channel becomes active at the same time, the one with the lower number is queued first. This mechanism provides a first-come, first-serve scheduling scheme between the concurrently active logical channels.

  In addition, each read and write port has a high-priority queue and a low-priority queue. The priority bits (WRITE_PRIORITY and READ_ PRIORITY) in the logical channel DMA4_CCRi register determine whether a logical channel is queued as high or low priority. A software-configurable 8-bit priority counter gives weighting to the priority write queue. For every N (1 to 255) schedules from the priority write queue, one is scheduled from the regular write queue. A channel that is scheduled goes to the end of the queue after it completes its turn on the port. The relative weighting of the scheduling of the high-priority queue to the low priority queue is programmable from 1:1 to 1:256 through the DMA global channel register using the DMA4_GCR[23:16] ARBITRATION_RATE bit field.

  NOTE: The DMA4_GCR[23:16] ARBITRATION_RATE bit field does not depend on the DMA4_GCR[13:12] HI_THREAD_RESERVED bit field. The ARBITRATION_RATE bit field depends on the DMA4_CCRi[26] WRITE_PRIORITY bit and the DMA4_CCRi[6] READ_PRIORITY bit.

- Read/write port access scheduling policy

  When either the read or write port becomes available, the port access scheduler selects the next logical channel for which to perform a DMA transaction from either the high- or low-priority queue.

  When the current DMA transaction (single or burst access) is complete and the full DMA transfer is not finished, the logical channel returns to the tail of the queue. Because the port access scheduling is on a per-transaction basis, a logical channel can be queued repeatedly this way several times during its block transfer.

The DMA_SYSTEM module can have up to four outstanding read transactions and two outstanding write transactions in the system interconnect; four read and two write thread IDs exist. For an arbitration cycle to occur, these two conditions must be met:

- At least one channel is requesting
- At least one free thread ID is available

On an arbitration cycle, the scheduler grants the highest priority channel that has an active request, allocates the thread ID, and tags this thread as busy. At a given time, a channel cannot be allocated for more than one thread ID.

NOTE: If more than one channel is active, each channel is given a thread ID for the current service only, not for the whole channel transfer.

When only one channel is enabled, only one thread is allocated for the channel. In such a situation the channel can have maximum of four outstanding commands (without getting the responses) without rescheduling the channel at the end of each transaction. Each command can be either single access (8-bit, 16-bit or 32-bit) or burst access (2 × M, 4 × M, 8 × M or 16 × M, where M can be 8, 16, or 32 bits).

When nonburst alignment is at the beginning of the transfer, the channel is rescheduled for each smaller access until burst-aligned. When the end of the transfer is not burst-aligned, the channel is rescheduled for each of the remaining smaller accesses.

For a logical channel transfer completion, when the last access is written to the destination, the logical channel becomes inactive. If enabled, an interrupt request is generated (see Section 16.1.4.2.1, *Interrupt Generation*).

16.1.4.4 FIFO Queue Memory Pool

A FIFO queue memory pool provides buffering between the read and write ports. The hardware allocates the space dynamically to a number of FIFO queues, and each queue is associated with an active logical channel.

To avoid a memory pool overflow, if there are fewer entries in the FIFO queue memory pool than are required for the maximum configured source burst size of the next logical channel to be scheduled, the logical channel is returned to the tail of the queue, and the port access scheduler continues to search the queue until it finds a logical channel that can be scheduled.

The maximum FIFO depth that can be allocated to each individual logical channel can be limited globally through the DMA4_GCR[7:0] MAX_CHANNEL_FIFO_DEPTH bit field. This value should be configured to allow a fair allocation of the memory pool between the active channels.

A logical channel is scheduled if it has not yet reached its allocation limit, even if the access to be performed will exceed this limit. This means that the effective number of entries used by a particular logical channel is limited to the configured maximum entries per channel + channel maximum configured burst size (in words) 1.

16.1.4.5 Addressing Modes

A DMA transfer block consists of a number of frames (FN). Each frame consists of a number of elements (EN), and each element can have a size of 8, 16, or 32 bits (ES), as follows:

transfer block size = FN × EN × ES

The FN, EN, and ES are common for the source and destination. However, the way in which the data is represented (addressing profile/mode) is independently programmable for the source and destination devices, using one of these four addressing modes:

- Constant: The address remains the same for consecutive element accesses.
- Post-increment: The address increases by the ES, even across consecutive frames.
- Single-index: The address increases by the ES plus the element index (EI) value minus 1 (even across consecutive frames).
- Double-index: The address increases by the ES plus the EI value minus 1 within a frame. When a full frame is transferred, the address increases by the ES plus the frame index (FI) value minus 1.

The ES, EI, and FI values are expressed in bytes. The EI and FI values can be positive or negative.

When calculating the EI and FI values, it is critical to note that, after an element is accessed, the logical channel address pointer equals the address of the last byte (highest address) of the accessed element. The correct value for the EI or FI must be such that, when added to the logical channel address pointer, it results in the address of the first byte (lowest address) of the next element to be accessed.

The EI and FI values must be configured so that the address of each element in the transfer is aligned on an ES boundary.

Consequently, the single-index addressing mode with EI = 1 or double-index addressing mode with EI = 1 and FI = 1 is equivalent to post-increment addressing.

NOTE: The source and destination start addresses must also be aligned on an ES boundary.

When the address of an element to be accessed is not aligned on an ES boundary, the transfer is stopped and a misaligned address error interrupt occurs, if enabled (see Section 16.1.4.2.1, *Interrupt Generation*).

The DMA4_CFNi register configures the FN in a block.

The DMA4_CENi register configures the EN.

The DMA4_CSDPi register configures the ES.

The DMA4_CSSAi and DMA4_CDSAi registers configure the source and destination start addresses.

The DMA4_CCRi register configures the source and destination addressing modes.

The DMA4_CSEIi, DMA4_CSFIi, DMA4_CDEIi, and DMA4_CDFIi registers configure the source EI, source FI, destination EI, and destination FI, respectively.

The addressing profiles are expressed as equations as follows:

Equation 1. Constant addressing:

A(n + 1) = A(n)

NOTE: Constant addressing mode with DMA4 to/from DDR memory is not supported on the device. To fill the DDR memory with a single value, the constant fill feature of the DMA4 must be used, instead of a constant-addressing mode transfer.

Equation 2. Post-increment addressing:

A(n + 1) = A(n) + ES

Equation 3. Single-indexed addressing:

A(n + 1) = A(n) + ES + (EI 1)

Equation 4. Double-indexed addressing:

When not at the end of a frame or transfer (that is, when the element counter ≠ 0):

A(n + 1) = A(n) + ES + (EI 1)

When at the end of a frame but not at the end of the transfer (that is, when the element counter = 0 and the frame counter ≠ 0):

A(n + 1) = A(n) + ES + (FI 1)

Calculate the element and frame index as follows:

Equation 5. Element index

EI = [(Stride EI 1) * ES] + 1

Equation 6. Frame index

FI = [(Stride FI 1) * ES] + 1

where:

A(n): Byte address of the element *n* within the transfer.

ES is in bytes, ES{1, 2, 4}.

EI is in bytes, specified in a configuration register, 32768 EI 32767.

Stride EI: The difference in the number of elements between the start of the current element *n* to the start of next element, *n+1*.

Element counter: A counter that is (re)initiated with the number of elements per frame or per transfer. Decreased by 1 for each element transferred. The initial value is configured in the register DMA channel element number, DMA4_CENi.

FI is in bytes, specified in a configuration register, 2147483648 FI 2147483647.

Stride FI: The difference in the number of elements between the start of the last element of the current frame and the beginning of the first element of the next frame.

Frame counter: A counter that is (re)initiated with the FN per transfer. Decreased by 1 for each frame transferred. The initial value is configured in the register DMA channel frame number, DMA4_CFNi.

Figure 16-6 shows how a stride EI and FI are defined. When handling complex configurations, using strides can make it easier to calculate EI and FI because you can calculate in elements instead of bytes. (This approach is used in the 90-degree clockwise image rotation example shown in Figure 16-10.) The double-index addressing example shown in Figure 16-6 uses ES = 4, EN = 2, EI = 5, FI = 5, and FN = 2.

Figure 16-6 through Figure 16-9 show examples of addressing mode configurations. Table 16-10 lists parameter values for the examples.

Figure 16-6. Example Showing Double-Index Addressing, Elements, Frames, and Strides

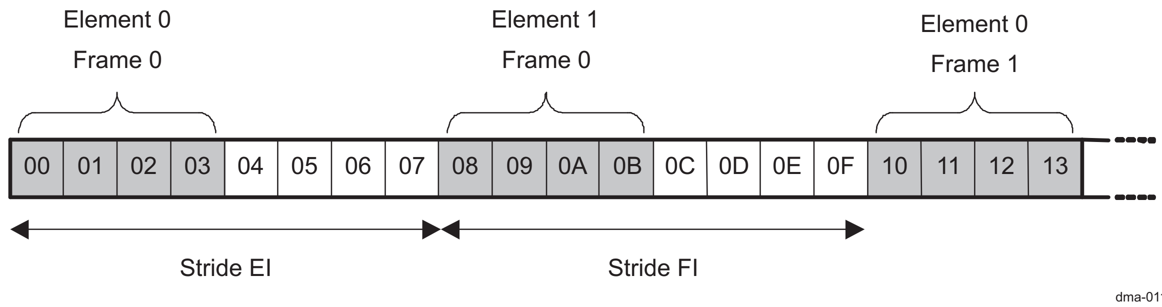


Figure 16-7. Addressing Mode Example (a)

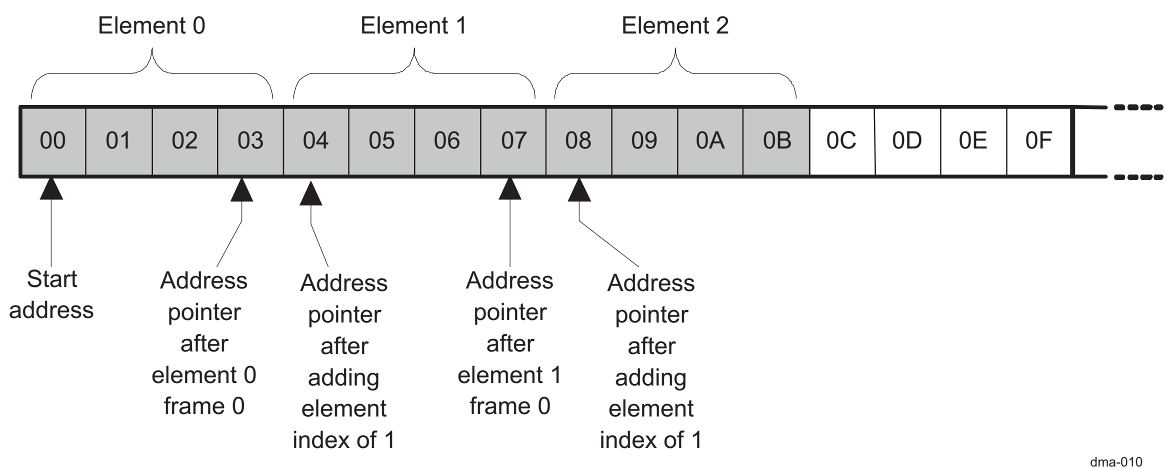


Figure 16-8. Addressing Mode Example (b)

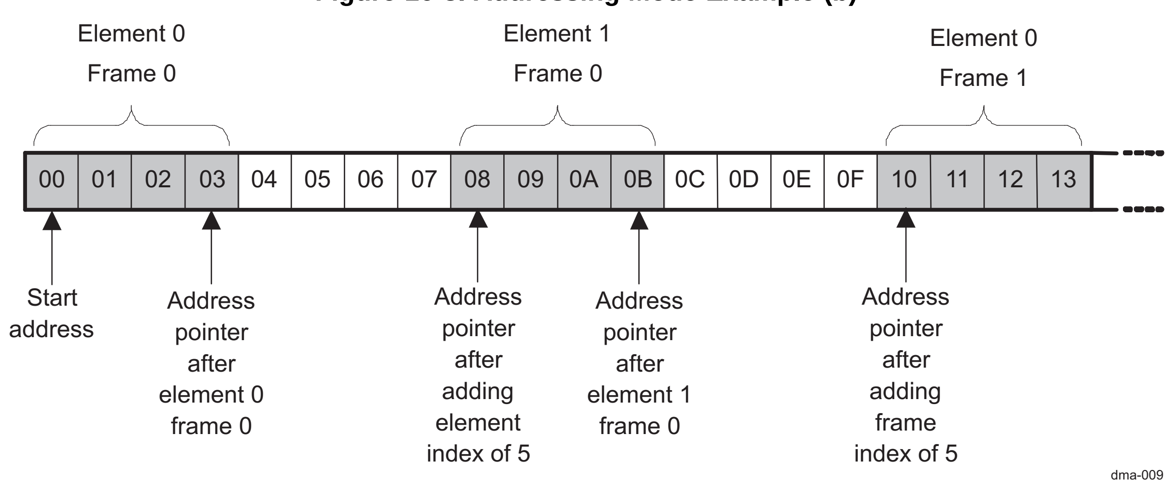


Figure 16-9. Addressing Mode Example (c)

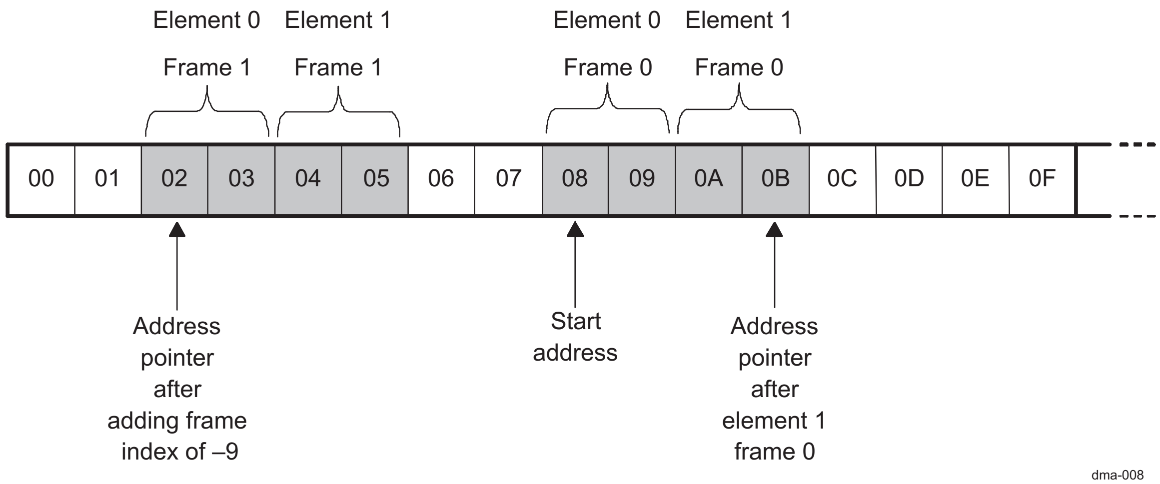


Table 16-10. Parameter Values for Addressing Mode Examples (a), (b), and (c)

Parameter	Example (a)	Example (b)	Example (c)
Addressing mode	Single index (or post-increment)	Double index	Double index
Start address	0	0	8
ES	4 (32-bit)	4 (32-bit)	2 (16-bit)
EN	3	2	2
EI	1	5	1
FN	1	2	2
Frame index	N/A	5	-9

Double indexing can occur on source (read) or destination (write). Equations for rotation of xx degrees on destination are obtained by taking equations for rotation of (360 – xx) degrees on source, and swapping the width (x) and height (y) of the image in them. The opposite is also true. Table 16-11 lists the equations for 90-, 180-, and 270-degree rotations.

Table 16-11. Equations for Rotation

		90° Rotation	180° Rotation	270° Rotation
Double indexing on destination (write)	Base address	ES*(y–1)	ES*(x*y–1)	ES*y*(x–1)
	EI	ES*(y–1) + 1	1–2*ES	1–ES*(y + 1)
	FI	1 ES*[(x–1)*y + 2]	1–2*ES	1+ES*(x–1)*y
Double indexing on source (read)	Base address	ES*x*(y–1)	ES*(x*y–1)	ES*(x–1)
	EI	1–ES*(x + 1)	1–2*ES	ES*(x–1) + 1
	FI	1+ES*(y–1)*x	1–2*ES	1 ES*[(y–1)*x + 2]

Table 16-12 and Figure 16-10 show the configuration required to perform a 90-degree clockwise rotation of a 240 × 160 pixel, 32-bit image. The EI, frame size, and FI values are configured so that the image is rotated line-by-line starting at the left end of the top line.

NOTE: The FI value for the destination is negative so that the first pixel of each subsequent line of the source image is written to the correct location at the destination.

Equation 5 and Equation 6 calculate the destination, FI and EI. The example assumes that the image lines are stored at consecutive addresses in memory, meaning that both EI and FI on the source side are 1.

Rotations:

Section 16.1.5.7, *90-Degree Clockwise Image Rotation*, describes how to program an example of a 90-degree clockwise image rotation.

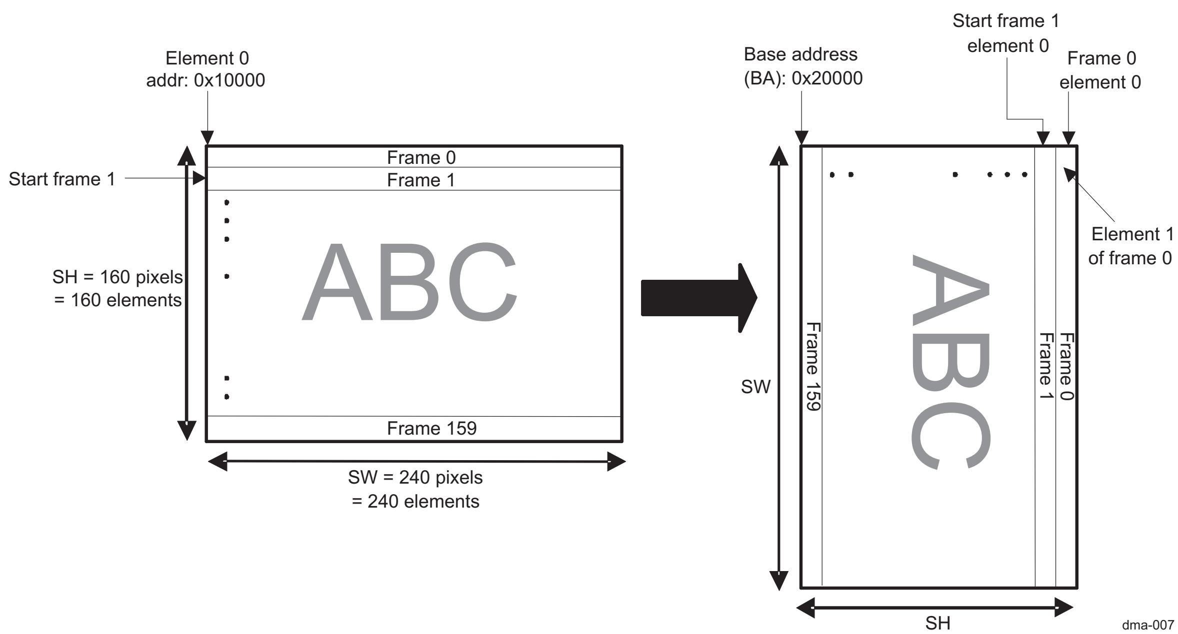
Observe that:

- One pixel = one element
- One line = one DMA frame
- Pixel size = element size = ES

Table 16-12. Example Parameter Values for a 90-Degree Clockwise Image Rotation

Parameter	Source Value	Destination Value
Bits per pixel	32	32
ES	4	4
Image width	SW	SH
Image height	SH	SW
Stride elements (stride EI)	1 element	SH
Stride frames (stride FI)	1 element	–[(SW-1)*SH + 1] = 38,241 elements
Start address	0x100000	0x200000 + (SH 1) x ES = 0x20027C
EN	SW	SW
EI	[(Stride EI 1) * ES] + 1 = 1	[(Stride EI 1) * ES] + 1 = 637
FN	SH	SH
FI	[(Stride FI 1) * ES] + 1= 1	[(Stride FI 1) * ES] + 1 = 152,967

Figure 16-10. Example of a 90-Degree Clockwise Image Rotation



16.1.4.6 Packed Accesses

To pack data means to group data to match the bus size, thus optimizing a transfer. When the logical channel ES is less than the DMA module read/write port size, and the addressing profile supports it (post-increment mode or single- or double-index mode with EI = 1), the number of elements to transfer in each read/write port access can be maximized by specifying that the source or destination is packed through the channel DMA4_CSDPi register. Thus:

- For a read/write port size of 32 bits, the source or destination can be configured as packed for transfer

ESs of 8 bits (four elements per access) and 16 bits (two elements per access).

- For a read/write port size of 64 bits, the source or destination can be configured as packed for transfer ESs of 8 bits (eight elements per access), 16 bits (four elements per access), and 32 bits (two elements per access).

Depending on the start address and transfer length, the first or last packed access can be only partially filled. This is indicated to the source or destination using the byte-enable signals.

16.1.4.7 Burst Transactions

Transfer performance can be improved so that the source or destination and addressing profile supports it. This can be achieved by configuring the logical channel to perform burst transactions consisting of multiple instead of single accesses. The channel can be programmed to use burst sizes equal to 16, 32, or 64 bytes through the DMA4_CSDPi register, with the read burst size programmable independently of the write burst size. Typically, the optimal burst size is 64 bytes (16 accesses for a 32-bit read/write port size or 8 accesses for a 64-bit read/write port size).

To obtain the maximum benefit from burst transactions, the source and destination start addresses must be aligned with the burst size. If this is not the case, the start of the transfer can consist of a number of smaller (single or burst) transactions until the first burst size boundary is reached.

Similarly, if the end of the transfer is not aligned on a burst size boundary, the final part of the transfer can consist of a number of smaller transactions.

NOTE: If post-incrementing is used, data must be packed to DMA data-port width, to use burst.

16.1.4.8 Endianism Conversion

The source and destination are each specified as little-endian or big-endian through the DMA4_CSDPi register for the particular logical channel. If the endianism of the source and destination differ, and if the logical channel ES is less than the DMA_SYSTEM module read/write port size, an endianism conversion is applied to the data before it is written to the destination.

When transferring data between a source and a destination with different endianism, it is important to specify an ES that equals the type of data being transferred to preserve the correct data image at the destination.

In the system, endianism conversion can be performed in more than one place. It is possible to instruct the source and/or destination to lock the endianism (that is, to not perform a conversion) through the logical DMA channel DMA4_CSDPi register.

NOTE: Because the device is little-endian by construction, the DMA_SYSTEM endianism registers must never be set to big-endian.

If DMA_SYSTEM is used to execute endian conversion by setting the source and destination to different endianism values, it is important to consider that the L3_MAIN interconnect also executes endian conversion if the DMA_SYSTEM and the source or destination have a different data bus width.

16.1.4.9 Transfer Synchronization

A logical channel can be programmed for software-triggered or hardware synchronized transfers.

16.1.4.9.1 Software Synchronization

A transfer is software-triggered when the logical channel is set up and started by software. To specify a software-triggered transfer, set the channel DMA DMA4_CCRi[4:0] and DMA4_CCRi[20:19] bit fields to 0. The transfer starts as soon as the DMA DMA4_CCRi[7] bit is set (when it enters the scheduling process).

16.1.4.9.2 Hardware Synchronization

A transfer is hardware-synchronized if the logical channel activation is driven by hardware requests from the source or destination target. A hardware-synchronized transfer is specified by configuring the DMA request line number in the channel DMA4_CCRi register to a value that corresponds to the DMA request line from the source or destination that generates the DMA requests. The DMA request numbers to be configured are specified in the DMA request mapping (see Table 16-9).

Specify the DMA request number in the DMA4_CCRi[4:0] SYNCHRO_CONTROL and DMA4_CCRi[20:19] SYNCHRO_CONTROL_UPPER bit fields. After the DMA4_CCRi[7] ENABLE bit is set, the logical channel becomes enabled but not activated (it does not enter the scheduling process), which means that channel registers are not updated until the first DMA request is received.

NOTE: The channel synchronization control registers are 1-based. For example, to enable the S_DMA_1 request, the DMA4_CCRi[4:0] SYNCHRO_CONTROL bit field must be set to 0x2 (DMA request number + 1).

NOTE: A DMA request line must not be shared between concurrently enabled DMA channels. However, a DMA request line can be shared among several chained logical channels.

For hardware synchronization, the amount of data to be transferred for each assertion of the DMA request line is configured through the frame synchronization (FS) and block synchronization (BS) bits in the logical channel DMA4_CCRi register and the DMA4_CCRi[5] FS and DMA4_CCRi[18] BS bits, respectively.

The amount of data can be any of the following:

- A single element transfer: A complete element defined by data type. For example, 8/16/32 bits are transferred in response to a DMA request.
- A full frame: A complete frame of several elements is transferred in response to a DMA request.
- A full block (a full channel transfer): A complete block of several frames is transferred in response to a DMA request.
- A full packet (a full channel transfer): A complete packet of several elements is transferred in response to a DMA request.

Packets allow the size of each part of the full DMA transfer to be configured independently of the organization of the data to be transferred (typically a number of elements). This can be useful when the source or destination has a buffer (such as a FIFO queue) with a size unrelated to the frame size of the transfer. The packet size then can be set to the size of the buffer.

Packet transfer must be used only where the source or destination is addressed in constant addressing mode, because FI registers are reused to specify the packet size.

To support the burst mode, the logical channel must also be configured in target-port packed access mode.

The packet size is configured based on the DMA4_CCRi[24] SEL_SRC_DST_SYNC bit through either the channel DMA4_CDFIi register (source synchronized) or the DMA4_CSFIi register (destination synchronized).

When the logical channel transfer block is not an exact multiple of the packet size, the final packet consists of the remaining elements in the transfer, using burst or single accesses to complete the block transfer.

The maximum transfer size, regardless of the packet size, is always as follows:

Block_size = Number_of_Frame_in_Block * Number_of_Element_in_Frame * Element_Size

- Synchronized at the source

  The DMA module optimizes the transfer with respect to the number and size of burst transactions for the given source and destination addressing profiles and configured maximum burst sizes. When writing to the destination is slower than reading from the source, data is buffered in the channel FIFO queue. If the transfer is packet-synchronized at the source, the end-of-packet interrupt is disabled (see Section 16.1.4.13, *Reprogramming an Active Channel*).

For a source synchronized transfer, buffering can be enabled or disabled by setting the DMA4_CCRi[25] BUFFERING_DISABLE bit. For a packet source synchronization with buffering disabled and the packed/burst across the packet boundary, the last packed/burst write transaction is split in optimized smaller accesses to complete the packet transfer size. However, for a packet source synchronized transfer with buffering enabled and with the packed/burst across the packet boundary, the DMA module waits for the next DMA request(s) to read enough data to issue an atomic packed/burst write transaction (assuming that the address is packed/burst aligned).

NOTE: Buffering is not performed between frames, even if it is enabled. If the packed/burst is across the frame boundary, the last packed/burst write transaction is split in optimized smaller accesses to complete the frame transfer size.

- Synchronized at the destination

  The performance of a hardware-synchronized transfer can be improved by using the prefetch mode, enabled through the channel DMA4_CCRi[23] PREFETCH bit. Data is prefetched on the read port side before the DMA request received and buffered in the FIFO queue. Up to a full transfer block can be prefetched, although this can be limited by the specified maximum channel FIFO queue depth (see Section 16.1.4.4, *FIFO Queue Memory Pool*).

  Buffering disable is not allowed for a destination-synchronized transfer.

NOTE: Behavior is undefined when prefetch is enabled and a transfer is synchronized to the source.

Regardless of whether buffering is enabled, the last transaction in the frame or in the block is write nonposted (WNP) even if the write mode is specified as write last nonposted (WLNP; the DMA4_CSDPi[17:16] WRITE_MODE bit field = 0x2). However, in a packet synchronization mode, the last transaction of each packet in the transfer is WNP only if the buffering disable is on (even if the write mode is specified as WLNP).

Regardless of whether buffering is enabled, the packet interrupt is not generated in the packet source synchronized mode.

CAUTION

The DMA4_CCRi[25] BUFFERING_DISABLE bit must be filled with an allowed value, as specified in Table 16-13.

Table 16-13. Buffering Disable

	BUFFERING_DISABLE (0: Buffering enable, 1: Buffering disable)	
Destination synchronized	0	Allowed
	1	Not allowed
Source synchronized	0	Allowed
	1	Allowed

- Synchronized transfer monitoring using CDAC (DMA4_CDACi):

  Context is restored only when the channel becomes active on a DMA request (not at software enable). The channel is software-enabled first, and then a DMA request is asserted followed by the first context restore.

  The CDAC register is writable; thus, the CDAC can be initialized to monitor the transfer and determine whether the transfer is started (for more information, see Section 16.1.5.4, *Synchronized Transfer Monitoring Using CDAC*).

NOTE: For 16-bit transactions, start reading from or writing to the LSByte first to enable the register update. This is not an issue for 32-bit read-write transactions.

16.1.4.10 Thread Budget Allocation

When several concurrent channels are latency critical and hardware synchronized, a specific latency cannot be ensured until the target is served. This situation occurs when the number of concurrent channels is greater than the number of available threads.

NOTE: Four threads are available on the read port, and two threads are available on the write port.

For a hardware-synchronized transfer (memory to peripheral), a minimum bandwidth for a latency-critical transfer must be ensured to avoid collisions between two hardware requests.

Because it is latency critical, the software user is responsible for the following:

- Programming the synchronized channel as a high-priority channel
- Reserving one or several threads for high-priority channels

The proposed implementation is as follows (see Section 16.1.5.5, *Concurrent Software and Hardware Synchronization*):

Prevent the regular channel queue from exceeding more than a programmable (3, 2, or 1) number of threads on the read port and no more than one thread on the write port. This number can be set in the global register DMA4_GCR[13:12] .

The thread reservation is programmable for maximum use of thread resources for concurrent, low-priority channel transfer. Programmability can also allow a partial throughput control by limiting in software the number of concurrent outstanding requests that break the pipelining.

Depending on the DMA4_GCR [13:12] value, the following threadID on the read/write ports are allocated for a high-priority channel:

Read port priority thread reservation:

- DMA4_GCR[13:12] = 0x0 => No ThreadID is reserved for high-priority channels.
- DMA4_GCR[13:12] = 0x1 => Read ThreadID 0 is reserved for high-priority channels.
- DMA4_GCR[13:12] = 0x2 => Read ThreadID 0 and Read ThreadID 1 are reserved for high-priority channels.
- DMA4_GCR[13:12] = 0x3 => Read ThreadID 0, Read ThreadID 1, and Read ThreadID 2 are reserved for high-priority channels.

Write port priority thread reservation:

- DMA4_GCR[13:12] = 0x0 => No ThreadID is reserved for high-priority channels
- DMA4_GCR[13:12] = 0x1 => Write ThreadID 0 is reserved for high-priority channels.
- DMA4_GCR[13:12] = 0x2 => Write ThreadID 0 is reserved for high-priority channels.
- DMA4_GCR[13:12] = 0x3 => Write ThreadID 0 is reserved for high-priority channels.

Regardless of whether the enabled channels are high priority, only the setting of the DMA4_GCR[13:12] value forces the thread reservation to these values. Set the appropriate value to avoid losing threads using only regular channels.

To have an independent read and write priority context, a per-channel bit (DMA4_CCRi[26]) is added for write priority, and the previous priority bit becomes read priority bit (DMA4_CCRi[6]).

NOTE: The device has one priority bit per logical channel, not one priority bit per port.

16.1.4.11 FIFO Budget Allocation

To avoid fully occupying the FIFO with a high-priority transfer while low-priority channels wait in the arbitration queue, two separate FIFO budgets are specified: one for high-priority channels and one for low-priority channels. This is defined in the DMA4_GCR register, allowing the user to share the FIFO budget between the low- and high-priority channels. The amount of the FIFO allocated by the low- and high-priority channels is fixed by the value set in the DMA4_GCR[15:14] HI_LO_FIFO_BUDGET field. The maximum channel FIFO depth is limited by the HI_LO_FIFO_BUDGET field as follows:

If the channel is low priority:

- When HI_LO_FIFO_BUDGET = 0x1, then low priority cannot exceed 75 percent of the total FIFO.
- When HI_LO_FIFO_BUDGET = 0x2, then low priority cannot exceed 25 percent of the total FIFO.
- When HI_LO_FIFO_BUDGET = 0x3, then low priority cannot exceed 50 percent of the total FIFO.

If channel is high priority

- When HI_LO_FIFO_BUDGET = 0x1, then high priority cannot exceed 25 percent of the total FIFO.
- When HI_LO_FIFO_BUDGET = 0x2, then high priority cannot exceed 75 percent of the total FIFO.
- When HI_LO_FIFO_BUDGET = 0x3, then high priority cannot exceed 50 percent of the total FIFO.

The user must perform the following equation:

- For a high-priority channel: (Per_Channel_Maximum FIFO Depth + 1) x Number of High Channel =< High Budget FIFO
- For a low-priority channel: (Per_Channel_Maximum FIFO Depth + 1) x Number of Low Channel =< Low Budget FIFO

NOTE: Ensure that *Number of High Channel* means *Number of Active High-Priority Channel* and that *Number of Low Channel* means *Number of Active Low-Priority Channel*.

16.1.4.12 Chained Logical Channel Transfers

Chaining multiple logical channels permits transfers consisting of multiple parts to be executed without repeated software intervention. This results in better performance than the alternative of software setting up and starting each transfer separately. Each part of a chained transfer can have the data addressed in a different manner that permits the programming of a variety of complex transfers. For example:

- Interlaced video data with one logical channel configured to transfer the even lines and another logical channel configured to transfer the odd lines
- Protocol headers with a separate DMA4 channel configured to transfer each field in the header

Channels can be chained through each channel DMA4_CLNK_CTRLi register. When the transfer for the first channel completes, the next channel in the chain is enabled. The number of channels in the chain that are configured for hardware-synchronized transfers is flexible (although typically it may be all, none, or simply the first one). The DMA request line number must be set to 0 to specify that any or all of the channels in a chain are software-triggered or nonsynchronized.

The last channel in a chain can be chained to the first channel to create a continuously looping chain. The continuously looping transfer can be stopped on the fly at a specific channel by disabling the DMA4_CLNK_CTRLi[15] ENABLE_LNK bit. The looping transfer stops after the specified channel transfer is complete.

NOTE: A DMA request line must not be shared between concurrently enabled DMA channels. However, a DMA request line can be shared between several chained logical channels.

For more information about the programming model, see Section 16.1.5, *DMA_SYSTEM Basic Programming Model*.

16.1.4.13 Reprogramming an Active Channel

A currently active logical DMA channel can be disabled through the DMA4_CCRi[7] ENABLE bit. When an ongoing transaction is complete and the read-active and write-active bits in the DMA4_CCRi register (DMA4_CCRi[9] RD_ACTIVE and DMA4_CCRi[10] WR_ACTIVE) are reset, the channel can be reprogrammed for a new transfer.

16.1.4.14 Packet Synchronization

A packet transfer notion is related to the behavior of some peripherals, which have certain buffering capability and requires to transfer the buffer content once an element number threshold is reached (a hardware DMA request is generated). To associate a frame synchronization to each DMA request is possible, but this limits the maximum transfer size. Indeed the maximum transfer size is proportional to the FIFO depth of the peripheral:

maximum_transfer_size =peripheral_FIFO_depth x number_of_frame_in_block

The packet synchronization allows to dissociate the transfer size from the FIFO depth of the peripheral. Only Constant addressing mode is allowed on a read port or a write port if source target or destination target is packet synchronized respectively.

Example:

Consider a camera interface with a FIFO_depth of 128 words and a FIFO_element_number_threshold of 128, and a picture to transfer with a size 320 lines by 240 columns. If frame synchronization is associated with each DMA request then the maximum transfer size that can be performed is $128x2^{16}$ words. In this case, a frame is 128-word long, which does not fit the size of a line, and it is not possible to generate an interrupt at the end of line. However, by introducing the packet transfer notion, which is related to the peripheral FIFO behavior/structure, the maximum transfer size (maximum_transfer_size = 2^{24} x 2^{16} words) is independent of both peripheral_FIFO_depth and FIFO_element_number_threshold. This allows a long-enough transfer within one channel context and rotation operation on a large image format.

The main features of DMA Packet transfer are as follows:

- DMA Packet_Data_Size for each DMA Request: The Peripheral_element_number_threshold (the number of elements in a packet) shares the DMA4_CSFIi and DMA4_CDFIi configuration registers. If the peripheral is the source target, the addressing mode is constant, and the DMA4_CSFIi[15:0] bit field is used to specify the packet data size in the DMA4_CSFIi register. The user must set the DMA.SDMA4_CCRi[24] SEL_SRC_DST_SYNC bit to 1. If the peripheral is the destination target, the addressing mode is constant, the DMA4_CDFIi[15:0], is used to specify the packet data size (PKT_ELNT_NBR), and the bit field [31:16] is unused. To specify the packet data size in the DMA4_CDFIi register, the user must set the DMA4_CCRi[24] SEL_SRC_DST_SYNC bit to 0.

NOTE: The packet size can be a submultiple or non-submultiple of a frame size. If DMA Packet_Data_Size is aligned on the DMA channel block data size boundary, then DMA transfers the last data in the channel block boundary and stops at the block boundary for the last packet DMA Request. If the Packet_Data_size is not aligned on the block boundary, the remaining data smaller than a packet size are transferred using burst or single accesses to complete the block.

- DMA Packet_Data_Transfer does not affect DMA channel capabilities in term of packing and bursting.

  The packet synchronization mode is active when DMA4_CCRi[5] FS = DMA4_CCRi[18] BS = 1. Then:

  - If DMA4_CCRi[24] SEL_SRC_DST_SYNC = 0, the DMA4_CDFIi[15:0] bit field gives the number of elements in the packet and the DMA4_CDFIi[31:16] bit field is unused for the packet size.
  - If DMA4_CCRi[24] SEL_SRC_DST_SYNC = 1, the DMA4_CSFIi[15:0] bit field gives the number of elements in the packet and the DMA4_CSFIi[31:16] bit field is unused for the packet size.

NOTE: The maximum transfer size, regardless of the packet size, is always:

Block_size = Number_of_Frame_in_Block x Number_of_Element_in_Frame x Element_Size.

If DMA channel packet/burst access is across the packet boundary, the DMA hardware automatically splits this packing/burst access into multiple smaller accesses that are aligned on the packet boundary. Otherwise, the DMA transfers data as a usual packing/burst access.

16.1.4.15 Graphics Acceleration Support

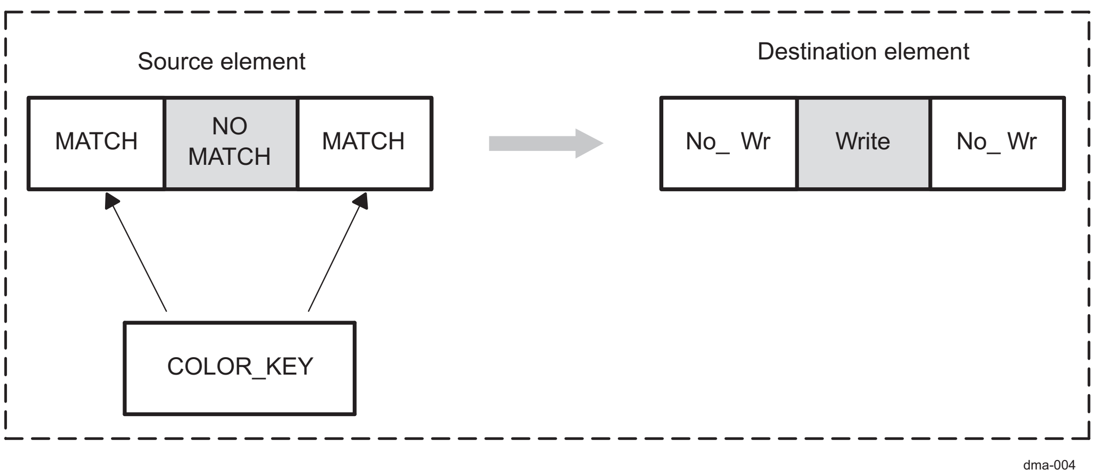
The DMA_SYSTEM supports two graphic acceleration features: transparent copy and constant fill.

Only one of these features can be enabled at a time through the DMA4_CCRi register for the particular logical DMA channel.

The transparent copy feature enables specification of a particular color through the DMA4_COLORi register so that when it is recognized in the data from the source, it is not copied to the corresponding location in the destination but instead leaves the data in the corresponding location in the destination as it is.

Figure 16-11 shows the 2-D graphic transparent color block diagram.

Figure 16-11. 2-D Graphic Transparent Color Block Diagram



The constant fill feature provides the ability to specify a particular color through the DMA4_COLORi register for every specified location in the destination. In this case, the transfer consists only of writing to the destination without reading from a source.

Both features support 8, 16, and 24 bpp, depending on what is specified as the DMA transfer ES through the DMA4_CSDPi register. An ES of 32 bits corresponds to 24 bpp. During a 32-bit (24 bpp) transfer, the 8 most-significant bits (MSBs) ([31:24]) are 0. Both features are compatible with packed and burst transactions.

16.1.4.16 Supervisor Modes

A logical DMA channel can be configured to operate in supervisor mode through the DMA4_CCRi[22] SUPERVISOR bit. This must be done using supervisor access. Once a channel is configured in supervisor mode, the channel configuration is protected from nonsupervisor accesses. All DMA transactions on a supervisor channel are supervisor transactions.

16.1.4.17 Posted and Nonposted Writes

A logical channel can be configured in its DMA4_CSDPi[17:16] bits to use one of three write access handshake modes for the destination:

- Nonposted write: Each write must complete before transfer can continue or complete.

- Posted write: Transfer continues without waiting for each write to complete (may improve performance with slow devices).
- Posted with final write nonposted: Transfer continues without waiting for each write to complete, but final write completes before transfer can complete.

16.1.4.18 Disabling a Channel During Transfer

When a channel is disabled during a transfer, the channel undergoes an abort, unless it is hardware-source-synchronized with buffering enabled (DMA4_CCRi[25] BUFFERING_DISABLE = 0). If this is the case, the FIFO is drained to prevent the loss of data. For more information about this feature, see Section 16.1.4.19, *FIFO Draining Mechanism*.

16.1.4.19 FIFO Draining Mechanism

When a source-synchronized channel is disabled during a transfer, the current hardware request (element/packet/frame/block) service completes and the channel DMA4_CCRi[9] RD_ACTIVE bit is set to 0, which means the channel is not active on the read port. The remaining data in the corresponding disabled channel FIFO is drained onto the write port and transferred to the programmed destination as in normal transfer.

At the end of the draining the DMA4_CCRi[10] WR_ACTIVE bit is set to 0 (channel is no longer active on the write port) and if the DMA4_CICRi[12] DRAIN_END_IE is set to 1, the DMA4_CSRi[12] DRAIN_END status bit is updated and an interrupt is generated.

Once a channel is disabled during a transfer, it must wait for the DMA4_CCRi[9] RD_ACTIVE and DMA4_CCRi[10] WR_ACTIVE bits to become 0 before being reenabled for a new transfer. The FIFO drain for a channel occurs only in the following cases:

- If the channel is a source synchronized channel and DMA4_CCRi[25] BUFFERING_DISABLE = 0
  and
- If the channel is not a solid fill channel
  and
- If the channel is not a transparent and copy channel
  and
- If the channel is a hardware, synchronized channel

NOTE: For a self-linked or chain-linked channel, the user must disable the DMA4_CLNK_CTRLi[15] ENABLE_LINK bit before disabling the channel.

In all other cases, the channel undergoes an abort.

16.1.4.20 Linked List

16.1.4.20.1 Overview

The DMA_SYSTEM supports the logical transfer-descriptor loader feature. A transfer descriptor represents a set of values that maps to a set of logical channel configuration registers.

A logical channel transfer descriptor can be loaded by DMA from memories, and then successive transfer descriptors can be autonomously loaded based on a linked-list scheme. This enables DMA4 scatter-gather transfers with minimum MPU support by removing successive channel configuration processing and associated interrupt handling overheads. It also optimizes DMA4 channel resources by enabling efficient transfer serialization on a single logical channel versus concurrent (multiple) logical channel use.

Different types of transfer descriptors are supported (full or partial logical channel configuration registers are set). This optimizes the memory size required for storing a long linked list, because parameter changes are limited to only a few logical channel configuration registers.

16.1.4.20.2 Link-List Transfer Profile

A linked-list transfer can be seen as a super-block transfer (where the block is composed of FN frames and each frame includes EN elements). The block size (FN x EN x ES) can be changed in the linked list by loading an updated transfer descriptor.

The end of the super block is signaled in the last descriptor associated with the last block. Generally, for a given link-list transfer, the logical channel is set at the beginning of the transfer and the logical channel configurations for the subsequent blocks are slightly changed. Thus, the descriptor can be limited to an update of only few parameters, such as FN or EN. This assumes that the content of unmodified registers is preserved when a new descriptor is loaded.

A transfer descriptor is composed of a set of channel configuration register values with the addition of the next-descriptor pointer register (DMA4_CNDPi) and a channel-descriptor parameter register (DMA4_CDPi). The next-descriptor pointer is the 32-bit address pointer from where the next transfer descriptor is to be loaded. The next-descriptor pointer is mapped depending on the descriptor type (1, 2, or 3).

16.1.4.20.3 Descriptors

A transfer descriptor is a set of values that maps to a set of logical channel configuration registers. The descriptor contains the parameters associated with a transfer profile (transfer size, source or destination addresses, etc). Four different types of transfer descriptors are supported to optimize the memory size required to store a long linked list and to minimize MPU use to create and maintain the descriptor list.

A transfer descriptor is a list of 32-bit values. A descriptor must be 32-bit aligned in memory. Only the 30 least-significant bits (LSBs) of the next-descriptor address pointer are updated from the descriptor, and the DMA4 forces the 2 LSBs to 0 on generation of the pointer address. The descriptor size is variable, depending on the descriptor type and the Nxt_Dv and Nxt_Sv bit fields.

Transfer descriptor bit mapping is the same as DMA4 logical-channel configuration register bit mapping, with the following exceptions:

- Src_Element_index and Dst_Element_index are concatenated in the same 32-bit location.
- DMA4_CICRi (interrupt event mask)
- CFN (frame number)
- Bit fields:
  - P: Corresponds to the PAUSE_LINK_LIST bit:
    - When set to 1 in the descriptor, the channel is suspended when the descriptor load completes.
    - The user must not set the PAUSE_LINK_LIST bit through the configuration port. Otherwise, behavior is undefined.
    - When set to 0 (through the configuration port) after pause, the linked-list channel resumes its transfer (descriptor load or data load).
  - B: Corresponds to the end-of-block enable bit (BLOCK_IE) of the DMA4_CICRi register; valid only for type 3. This value is don't care for descriptor types 1 and 2, where DMA4_CICRi is fully specified.
  - Nxt_Dv, Nxt_Sv: Mapped in the DMA4_CDPi register. They indicate one of the following possibilities:
    - Next descriptor contains an updated destination or source address.
    - Next descriptor does not update the source or destination address, but increments the last source or destination address (from the end of the last transfer).
    - The next source address and/or destination address are the last valid ones in the configuration memory. This means that the corresponding location in the configuration memory is not updated (assuming that they were initialized at least once in the past). This is also called wrapping addressing.
  - Next_Descriptor_Type: Specifies the next descriptor type that corresponds to the NEXT_DESCRIPTOR_TYPE bit field in the DMA4_CDPi register

16.1.4.20.3.1 Type 1

A type 1 descriptor includes the overall channel configuration register value to be loaded (global registers are not part of the type 1 descriptor). This descriptor is used primarily when major changes are required:

- Channel read or write access profiles must be modified; for example, bursting and packing (included in the DMA4_CSDPi register)
- Attach a new DMA request to the same channel or change the priority or access privilege (included in the DMA4_CCRi register)
- Enable solid or transparent color fill (included in the DMA4_CCRi and DMA4_COLORi registers)
- Enable a channel link (included in the DMA4_CLNK_CTRLi register)

Table 16-14 shows a type 1 descriptor.

Table 16-14. Type 1

<table>
<tr><th></th><th>31</th><th>30</th><th>29</th><th>28</th><th>27</th><th>26</th><th>25</th><th>24</th><th>23</th><th>22</th><th>21</th><th>20</th><th>19</th><th>18</th><th>17</th><th>16</th><th>15</th><th>14</th><th>13</th><th>12</th><th>11</th><th>10</th><th>9</th><th>8</th><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td>Ptr+ 0x2C</td><td colspan="32">CCR</td></tr>
<tr><td>Ptr+ 0x28</td><td colspan="32">CLNK_CTRL</td></tr>
<tr><td>Ptr+ 0x24</td><td colspan="32">CSDP</td></tr>
<tr><td>Ptr+ 0x20</td><td colspan="32">COLOR</td></tr>
<tr><td>Ptr+ 0x1C</td><td colspan="32">Src_Frame_index/Src_Packet_size</td></tr>
<tr><td>Ptr+ 0x18</td><td colspan="32">Dst_Frame_index/Dst_Packet_size</td></tr>
<tr><td>Ptr+ 0x14</td><td colspan="16">Src_Element_index</td><td colspan="16">Dst_Element_index</td></tr>
<tr><td>Ptr+ 0x10</td><td colspan="16">CICR (interrupt events mask)</td><td colspan="16">CFN frame number</td></tr>
<tr><td>Ptr+ 0xC</td><td colspan="32">Destination_Start_Address</td></tr>
<tr><td>Ptr+ 0x8</td><td colspan="32">Source_Start_Address</td></tr>
<tr><td>Ptr+ 0x4</td><td colspan="3">N_type</td><td>B</td><td colspan="2">Dv</td><td colspan="2">Sv</td><td colspan="24">Element_number</td></tr>
<tr><td>Ptr</td><td colspan="30">Next_descriptor_address_pointer</td><td>Rsv</td><td>P</td></tr>
</table>

16.1.4.20.3.2 Type 2

A type 2 descriptor includes the overall logical channel transfer address register and transfer format register to be loaded. This descriptor enables 2D addressing linked-list transfer (for example, a multimedia application where 2D objects are moved in a link). Table 16-15 shows a type 2 descriptor with source and destination address updates. Table 16-16 shows a type 2 descriptor with one source or destination address update.

Table 16-15. Type 2 With Source and Destination Address Updates

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Ptr+ 0x1C	Src_Frame_index/Src_Packet_size																															
Ptr+ 0x18	Dst_Frame_index/Dst_Packet_size																															
Ptr+ 0x14	Src_Element_index																Dst_Element_index															
Ptr+ 0x10	CICR (interrupt events Mask)																CFN frame number															
Ptr+ 0xC	Destination_Start_Address																															
Ptr+ 0x8	Source_Start_Address																															
Ptr+ 0x4	N_type			B	Dv		Sv		Element_number																							
Ptr	Next_descriptor_address_pointer																														Rsv	P

Table 16-16. Type 2 With Source or Destination Address Update

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Ptr+ 0x18	Src_Frame_index/Src_Packet_size																															
Ptr+ 0x14	Dst_Frame_index/Dst_Packet_size																															
Ptr+ 0x10	Src_Element_index																Dst_Element_index															
Ptr+ 0xC	CICR (interrupt events Mask)																CFN frame number															
Ptr+ 0x8	Source_Start_Address or Destination_Start_Address																															
Ptr+ 0x4	N_type			B	Dv		Sv		Element_number																							
Ptr	Next_descriptor_address_pointer																														Rsv	P

16.1.4.20.3.3 Type 3

A type 3 descriptor is limited to a few logical channel transfer address registers and transfer format registers to be loaded. This descriptor enables simple 1D addressing link transfer (for example, scatter-gather or ping-pong memory movement using a linked list). Table 16-17 shows a type 3 descriptor with source and destination address updates. Table 16-18 shows a type 3 descriptor with one source or address destination update.

Table 16-17. Type 3 With Source and Destination Address Updates

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Ptr+ 0xC	Destination_Start_Address																															
Ptr+ 0x8	Source_Start_Address																															
Ptr+ 0x4	N_type			B	Dv		Sv		Element_number																							
Ptr	Next_descriptor_address_pointer																														Rsv	P

Table 16-18. Type 3 With Source or Destination Address Update

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Ptr+ 0x8	Source_Start_Address or Destination_Start_Address																															
Ptr+ 0x4	N_type			B	Dv		Sv		Element_number																							
Ptr	Next_descriptor_address_pointer																														Rsv	P

16.1.4.20.4 Linked-List Control and Monitoring

16.1.4.20.4.1 Transfer Mode Setting

Four descriptor types are available in DMA4_CDPi[9:8] TRANSFER_MODE to distinguish the different transfer modes:

- DMA4_CDPi[9:8] TRANSFER_MODE = 00: The current channel is using normal mode.
- DMA4_CDPi[9:8] TRANSFER_MODE = 01: The current channel is using link-list channel mode for a type 1, 2, or 3 descriptor.

The reset value is normal mode (DMA4_CDPi[9:8] TRANSFER_MODE = 0).

16.1.4.20.4.2 Starting a Linked List

Like a nonlinked-list transfer, a link transfer starts under host control by enabling the associated logical channel (set the DMA4_CCRi[7] ENABLE bit to 1). The DMA4_CDPi[10] FAST bit sets the start mode of the link-list transfer:

In nonfast-start mode, the logical channel configuration is fully initialized so that the transfer can start without descriptor loading.

In fast-start mode, the descriptor pointer and other inputs are given. The channel starts by loading the descriptor and then starts the data transfer phase.

16.1.4.20.4.3 Monitoring a Linked-List Progression

In addition to the DMA4_CCENi (remaining elements) and DMA4_CCFNi (remaining frames) registers that are used to monitor the transfer progress, a per-channel register, DMA4_CCDNi (channel current active descriptor number), monitors which descriptor in the list is active. The user must initialize the DMA4_CCDNi register to 0 during the initial configuration. When the DMA4_CCDNi register is updated, the DMA4_CCFNi and the DMA4_CCENi registers are updated. The user must also initialize the DMA4_CCFNi and DMA4_CCENi registers to 0xFFFF and to 0xFFFFFF, respectively, to track the effective transfer start of synchronized transfer.

16.1.4.20.4.4 Interrupt During Linked-List Execution

Any logical channel source of interrupt can be triggered during a linked-list execution, if the interrupt source is enabled during the initial configuration in CICR. The DMA4_CICRi register can also be updated during the linked-list execution if descriptor types 1 and 2 are used.

The use of an interrupt event in a link execution can be difficult, because the link can progress in parallel with interrupt service routine (ISR) execution. This makes it difficult to synchronize them unless system assumptions are used. The most appropriate synchronization model is to get an interrupt-only on linked-list completion, when the last transfer block is complete. This prevents the interrupt from occurring during the link execution. An end-of-super-block interrupt event available in the DMA4_CICRi and DMA4_CSRi registers can be enabled at initial configuration or when using descriptor types 1 and 2. To prevent the use of descriptor type 1 or 2 to update BLOCK_IE (full DMA4_CICRi update), a dedicated BLOCK_IE bit field is also available in a type 3 descriptor.

16.1.4.20.4.5 Pause a Linked List

When the channel is suspended, it remains enabled.

The pause behaves differently, depending on the transfer mode:

- Normal transfer mode: If the user sets the DMA4_CDPi[7] PAUSE_LINK_LIST bit to 1, the channel completes the current read and write transactions and then suspends the channel. The channel can be resumed by setting the channel DMA4_CDPi[7] PAUSE_LINK_LIST bit to 0.
- Linked-list type 1, 2, or 3 mode: The user must not set the DMA4_CDPi[7] PAUSE_LINK_LIST bit through the configuration port; otherwise, transfer behavior is undefined.

  A PAUSE_LINK_LIST bit (P) is set to 1 in the descriptor.
  - The channel is suspended after the descriptor load, translation, and configuration memory update are complete.
  - The linked list can be resumed by resetting the DMA4_CDPi[7] PAUSE_LINK_LIST bit (through the configuration port).

16.1.4.20.4.6 Stop a Linked List (Abort or Drain)

The channel can be stopped for a drain or an abort. These cases are exclusive.

16.1.4.20.4.6.1 Drain

- Drain conditions:

  A channel is a drain candidate if it is a hardware-source-synchronized transfer with DMA4_CCRi[25] BUFFERING_DISABLE = 0 and should not be doing any of the graphics operation (transparent copy or solid-color fill).
- Drain trigger:

  A drain candidate channel is drained if it is disabled (DMA4_CCRi[7] ENABLE = 0) or if it receives a transaction error on the read port.
- Drain behavior with a type 1, 2, or 3 descriptor. Drain trigger can occur in two situations:
  - During descriptor loading: Any ongoing current transaction is complete and the channel is aborted.
  - During data loading: The read is completed at the boundary of the request (element/frame/packet/block boundary), the FIFO is drained to the destination, and then a DRAIN_END interrupt can be asserted.

16.1.4.20.4.6.2 Abort

- Abort condition:

  A channel is an abort candidate if it is software-synchronized, hardware-destination-synchronized, solid color-fill, transparent-color fill, or hardware-source-synchronized with DMA4_CCRi[25] BUFFERING_DISABLE = 1.
- Abort trigger:

  A channel is an abort candidate if it is disabled (DMA4_CCRi[7] ENABLE = 0), if it receives a transaction error on the read or write port, or if there is a MISALIGNMENT_ERROR.
- Abort behavior with a type 1, 2, or 3 descriptor:

  If an abort trigger occurs, the channel aborts immediately after completion of current read/write transactions and then the FIFO is cleaned up.

  In type 1, 2, or 3, if an abort trigger or drain trigger occurs during the descriptor load phase, the channel aborts.

16.1.4.20.4.7 Status Bit Behavior

This section describes the behavior of the DMA4_CSRi[6] SYNC, DMA4_CCRi[9] RD_ACTIVE and DMA4_CCRi[10] WR_ACTIVE status bits:

- For a hardware-synchronized channel in linked-list mode, the DMA4_CSRi[6] SYNC bit becomes active (DMA4_CSRi[6] SYNC = 1) when the first data load transaction is scheduled and remains active until the last data load transaction in the block (not super block) is descheduled (DMA4_CSRi[6] SYNC = 0). The SYNC bit is not active during the descriptor load phase.

- The DMA4_CCRi[9] RD_ACTIVE bit is active during the data load phase and the descriptor load phase. It becomes active when the first read transaction is scheduled. It becomes inactive:
  – When (during the descriptor load phase) the last descriptor write request is descheduled
  – When (during the data load phase) the last read transaction in the block (not super block) is descheduled for software-synchronized transfer or destination-synchronized transfer with prefetch enabled
  – When (during the data load phase) the last read transaction in the request (element/frame/packet/block sync) is descheduled for hardware-source-synchronized transfer or hardware-destination-synchronized transfer without prefetch
- The DMA4_CCRi[10] WR_ACTIVE bit is active only during the data load phase. It becomes active when the first write transaction is scheduled and becomes inactive:
  – Until the last write transaction in the block (not super block) is descheduled and the FIFO is cleaned up for software-synchronized transfer
  – Until the last write transaction in the request (element/frame/packet/block sync) is descheduled and the FIFO is cleaned up for hardware-source-synchronized transfer (with DMA4_CCRi[25] BUFFERING_DISABLE = 0) or hardware-destination-synchronized transfer.

16.1.4.20.4.8 Linked-List Channel Linking

Channel linking for inter- and intra-super blocks is supported for type 1, 2, and 3 descriptors.

Assume that CHx and CHz are linked-list channels using generic descriptors. If CHx is composed of N descriptors and CHz is composed of M descriptors, then in nonfast mode:

CHx: CHx[Data1]-> CHx[DES1] -> . -> CHx[DESN]->CHx[DataN + 1]

CHz: CHz[Data1]-> CHz[DES1] -> . -> CHz[DESM]->CHz[DataM + 1]

It is possible to link CHx to CHz or CHx to itself after the completion of the CHx transfer (end of super block). To do this, the user must set the DMA4_CLNK_CTRLi[15] ENABLE_LNK bit to 1 and the DMA4_CLNK_CTRLi[4:0] NEXTLCH_ID bit to z (or to x for self linking) through the last descriptor using a type 1 descriptor. The sequence is:

CHx: CHx[Data1]-> CHx[DES1] -> . -> CHx[DESN]-CHx[DataN+1] -> CHz: CHz[Data1]-> CHz[DES1] -> . -> CHz[DESM]->CHz[DataM+1]

It is also possible to link CHx to CHz during the CHx transfer and before the end of super block. The user must set the DMA4_CLNK_CTRLi[15] ENABLE_LNK bit to 1 and the DMA4_CLNK_CTRLi[4:0] NEXTLCH_ID bit to z through descriptor p (CHx[DESp]) using a type 1 descriptor. The sequence is:

CHx: CHx[Data1]-> CHx[DES1] ->.-> CHx[DESp]->CHx[Data(p + 1)] -> CHz[Data1]-> CHz[DES1] -> .

The user must continue the linking until channels CHx and CHz complete their super-block transfers; otherwise, the channels remain enabled.

NOTE: In channel linking, the head of a chain can be in fast mode or nonfast mode. All channels that are not in the head of the chain can be in nonfast mode only. In self-linking, the channel cannot be in fast mode.

NOTE: If channel CHx links to CHz in the middle of the superblock transfer (remember link bit can be set through Type-1 descriptor load), CHx is disabled after the corresponding data load and enables the channel CHz.

16.1.5 DMA_SYSTEM Basic Programming Model

16.1.5.1 Setup Configuration

After a hardware reset, program all fields in the logical channel registers to default values for any channels used, because most fields are undefined following reset.

Before programming any DMA transfers, the priority arbitration rate and the maximum FIFO depth must be configured through the DMA4_GCR register, and any required interrupts must be enabled through the DMA4_IRQENABLE_Lj registers and the logical channel DMA4_CICRi registers.

Software clears the DMA4_CSRi register and the IRQSTATUS bit for the different interrupt lines before enabling the channel.

16.1.5.2 Software-Triggered (Nonsynchronized) Transfer

To program a software-triggered DMA transfer:

1. Configure the transfer parameters in the logical DMA channel registers:
   - DMA4_CSDPi:
     - Transfer ES (8, 16, or 32 bits) in the DMA DMA4_CSDPi[1:0] bit field.
     - Read and write port access types (single/burst), DMA DMA4_CSDPi[8:7] and DMA4_CSDPi[15:14] bit fields
     - Source and destination endianism, DMA DMA4_CSDPi[21] and DMA4_CSDPi[19] bits
     - Write mode (posted or nonposted) and DMA DMA4_CSDPi[17:16] bit field
     - Source or destination packed or nonpacked (if the ES is less than the read/write port size), DMA DMA4_CSDPi[6] and DMA4_CSDPi[13] bits
   - DMA4_CENi: EN
   - DMA4_CFNi: FN per transfer block
   - DMA4_CSSAi and DMA4_CDSAi: Source and destination start address (aligned with transfer ES)
   - DMA4_CCRi:
     - Read and write port addressing modes, DMA DMA4_CCRi[13:12] and DMA4_CCRi[15:14] bit field
     - Priority bit for both read and write ports, DMA DMA4_CCRi[6] and DMA4_CCRi[26] bits
     - DMA request number (set to 0 for a software-triggered transfer) and DMA register bit fields DMA4_CCRi[4:0] = 0 and DMA4_CCRi[20:19] = 0
   - DMA4_CSEIi, DMA4_CSFIi, DMA4_CDEIi, and DMA4_CDFIi: Source and destination element and frame indexes (depending on addressing mode)
2. Start the transfer through the enable bit in the channel DMA4_CCRi register and DMA DMA4_CCRi[7] bit

The following example performs a DMA transfer on channel 10 of a 240*160 picture from RAM to RAM (0x80C00000 to 0x80F00000):

```
UWORD32 RegVal = 0;
DMA4_t *DMA4;
DMA4 = (DMA4_t
      *)malloc(sizeof(DMA4_t));

/* Init. parameters
      */
DMA4->DataType = 0x2; //
      DMA4_CSDPi[1:0]
DMA4->ReadPortAccessType = 0; //
      DMA4_CSDPi[8:7]
DMA4->WritePortAccessType = 0; //
      DMA4_CSDPi[15:14]
DMA4->SourceEndiansim = 0; //
      DMA4_CSDPi[21]
DMA4->DestinationEndianism = 0; //
```

```
        DMA4_CSDPi[19]
  DMA4->WriteMode = 0; //
        DMA4_CSDPi[17:16]
  DMA4->SourcePacked = 0; //
        DMA4_CSDPi[6]
  DMA4->DestinationPacked = 0; //
        DMA4_CSDPi[13]
  DMA4->NumberOfElementPerFrame = 240; //
        DMA4_CENi
  DMA4->NumberOfFramePerTransferBlock = 160; //
        DMA4_CFNi
  DMA4->SourceStartAddress = 0x80C00000; //
        DMA4_CSSAi
  DMA4->DestinationStartAddress = 0x80F00000; //
        DMA4_CDSAi
  DMA4->SourceElementIndex = 1; //
        DMA4_CSEIi
  DMA4->SourceFrameIndex = 1; //
        DMA4_CSFIi
  DMA4->DestinationElementIndex = 1; //
        DMA4_CDEIi
  DMA4->DestinationFrameIndex = 1; //
        DMA4_CDFIi
  DMA4->ReadPortAccessMode = 1; //
        DMA4_CCRi[13:12]
  DMA4->WritePortAccessMode = 1; //
        DMA4_CCRi[15:14]
  DMA4->ReadPriority = 0; //
        DMA4_CCRi[6]
  DMA4->WritePriority = 0; //
        DMA4_CCRi[23]
  DMA4->ReadRequestNumber = 0; //
        DMA4_CCRi[4:0]
  DMA4->WriteRequestNumber = 0; //
        DMA4_CCRi[20:19]


  /* 1) Configure the transfer
        parameters in the logical DMA registers
        */
  /*------------------------------------------------------------------*/

  /*
        a) Set the data type CSDP[1:0], the Read/Write Port access type
        CSDP[8:7]/[15:14], the Source/dest endianism CSDP[21]/CSDP[19], write
        mode CSDP[17:16], source/dest packed or non-packed
        CSDP[6]/CSDP[13]*/

  // Read CSDP
  RegVal =
        DMA4_CSDP_CH10;

  // Build reg
  RegVal = ((RegVal &~ 0x3)
        | DMA4->DataType );
  RegVal = ((RegVal &~(0x3 << 7)) |
        (DMA4->ReadPortAccessType << 7));
  RegVal = ((RegVal &~(0x3 << 14)) |
        (DMA4->WritePortAccessType << 14));
  RegVal = ((RegVal &~(0x1 << 21)) |
        (DMA4->SourceEndiansim << 21));
  RegVal = ((RegVal &~(0x1 << 19)) |
        (DMA4->DestinationEndianism << 19));
  RegVal = ((RegVal &~(0x3 << 16)) |
        (DMA4->WriteMode << 16));
  RegVal = ((RegVal &~(0x1 << 6)) |
```

```
        (DMA4->SourcePacked << 6));
  RegVal = ((RegVal &~(0x1 << 13)) |
        (DMA4->DestinationPacked << 13));

  // Write CSDP

  DMA4_CSDP_CH10 = RegVal;

  /* b) Set the number of
        element per frame CEN[23:0]*/

  DMA4_CEN_CH10 =
        DMA4->NumberOfElementPerFrame;

  /* c) Set the number of frame
        per block CFN[15:0]*/

  DMA4_CFN_CH10 =
        DMA4->NumberOfFramePerTransferBlock;

  /* d) Set the
        Source/dest start address index CSSA[31:0]/CDSA[31:0]*/


  DMA4_CSSA_CH10 = DMA4->SourceStartAddress; // address start

  DMA4_CDSA_CH10 = DMA4->DestinationStartAddress; // address dest


  /* e) Set the Read Port addressing mode CCR[13:12], the
        Write Port addressing mode CCR[15:14], read/write priority
        CCR[6]/CCR[26], the current LCH CCR[20:19]=00 and CCR[4:0]=00000*/


  // Read CCR
  RegVal = DMA4_CCR_CH10;

  //
        Build reg
  RegVal = ((RegVal &~(0x3 << 12)) | (DMA4->ReadPortAccessMode      << 12));
  RegVal = ((RegVal &~(0x3 << 14)) | (DMA4->WritePortAccessMode
        << 14));
  RegVal = ((RegVal &~(0x1 << 6)) | (DMA4->ReadPriority << 6));

  RegVal = ((RegVal &~(0x1 << 26)) | (DMA4->WritePriority << 26));

  RegVal &= 0xFFCFFFE0 ;

  // Write CCR
  DMA4_CCR_CH10
        = RegVal;

  /* f)- Set the source element index CSEI[15:0]*/


  DMA4_CSEI_CH10 = DMA4->SourceElementIndex;

  /* g)-
        Set the source frame index CSFI[15:0]*/

  DMA4_CSFI_CH10 =
        DMA4->SourceFrameIndex ;

  /* h)- Set the destination element
        index CDEI[15:0]*/
```

```
DMA4_CDEI_CH10 =
      DMA4->DestinationElementIndex;

/* i)- Set the destination
      frame index CDFI[31:0]*/

DMA4_CDFI_CH10 =
      DMA4->DestinationFrameIndex;

/* 2) Start the DMA transfer by
      Setting the enable bit CCR[7]=1 */

/*--------------------------------------------------------------*/


//write enable bit
DMA4_CCR_CH10 |= 1 << 7; /* start */
```

16.1.5.3 Hardware-Synchronized Transfer

To monitor a hardware synchronized DMA transfer, initialize the DMA4_CDACi register before the software enable.

To configure an LCh to synchronize by element, packet, frame, or block, the frame synchronization DMA4_CCRi[5] FS bit and the block synchronization DMA4_CCRi[18] BS bit must be programmed. For all the following synchronized transfers (element, packet, and frame or block-synchronized transfers), the user must first set the DMA4_CCRi[24] SEL_SRC_DST_SYNC bit to 1 when the source triggers on the DMA request and set it the DMA4_CCRi[24] SEL_SRC_DST_SYNC bit to 0 when the destination triggers on the DMA request.

NOTE: The user must take care when setting the DMA4_CCRi[23] PREFETCH bit it is in conjunction with DMA4_CCRi[24] SEL_SRC_DST_SYNC bit.

- To configure an LCh to transfer one element per DMA request:
  1. Set the number of DMA request associated with the current LCH in the DMA4_CCRi[20:19] SYNCHRO_CONTROL_UPPER and DMA4_CCRi[4:0] SYNCHRO bit field.
  2. Set the data type, also referenced as element size (ES), in the DMA4_CSDPi[1:0] DATA_TYPE bit field.
  3. Set the Read Port access type (single or burst access) in the DMA4_CSDPi[8:7] SRC_BURST_EN bit field.
  4. Set the Write Port access type (single or burst access) in the DMA4_CSDPi[15:14] DST_BURST_EN bit field.
  5. Set the Read Port addressing mode in the DMA4_CCRi[13:12] SRC_AMODE bit field.
  6. Set the Write Port addressing mode in the DMA4_CCRi[15:14] DST_AMODE bit field.
  7. Set the Read start address in the DMA4_CSSAi[31:0] SRC_START_ADRS bit field.
  8. Set the Write start address in the DMA4_CDSAi[31:0] DST_START_ADRS bit field.
  9. Set both FS and BS to 0 in DMA4_CCRi[5] FS and DMA4_CCRi[18] BS.
  10. Set to 1 the channel enable bit DMA4_CCRi[7] EN.
- To configure an LCh to transfer one frame per DMA request:
  1. Set the number of DMA request associated to the current LCH in the DMA4_CCRi[20:19] SYNCHRO_CONTROL_UPPER and DMA4_CCRi[4:0] SYNCHRO bit field.
  2. Set the data type, also referenced as element size (ES), in the DMA4_CSDPi[1:0] DATA_TYPE bit field.
  3. Set the number of element per frame in the DMA4_CENi[23:0] CHANNEL_ELMNT_NBR bit field.
  4. Set the Read Port access type (single or burst access) in the DMA4_CSDPi[8:7] SRC_BURST_EN bit field.

5. Set the Write Port access type (single or burst access) in the DMA4_CSDPi[15:14] DST_BURST_EN bit field.
6. Set the Read Port addressing mode in the DMA4_CCRi[13:12] SRC_AMODE bit field.
7. Set the Write Port addressing mode in the DMA4_CCRi[15:14] DST_AMODE bit field.
8. Set the Read start address in the DMA4_CSSAi[31:0] SRC_START_ADRS bit field.
9. Set the Write start address in the DMA4_CDSAi[31:0] DST_START_ADRS bit field.
10. Set FS to 1 and BS to 0, respectively, in DMA4_CCRi[5] FS and DMA4_CCRi[18] BS.
11. Set to 1 the channel enable bit DMA4_CCRi[7] EN.

- To configure an LCh to transfer one block per DMA request:
  1. Set the number of DMA request associated to the current LCH in the DMA4_CCRi[20:19] SYNCHRO_CONTROL_UPPER and DMA4_CCRi[4:0] SYNCHRO bit field.
  2. Set the data type, also referenced as element size (ES), in the DMA4_CSDPi[1:0] DATA_TYPE bit field.
  3. Set the number of element per frame in the DMA4_CENi[23:0] CHANNEL_ELMNT_NBR bit field.
  4. Set in the DMA4_CFNi[15:0] CHANNEL_FRAME_NBR bit field the number of frame (transfers), to take place before the LCH gets disabled.
  5. Set the Read Port access type (single or burst access) in the DMA4_CSDPi[8:7] SRC_BURST_EN bit field.
  6. Set the Write Port access type (single or burst access) in the DMA4_CSDPi[15:14] DST_BURST_EN bit field.
  7. Set the Read Port addressing mode in the DMA4_CCRi[13:12] SRC_AMODE bit field.
  8. Set the Write Port addressing mode in the DMA4_CCRi[15:14] DST_AMODE bit field.
  9. Set the Read start address in the DMA4_CSSAi[31:0] SRC_START_ADRS bit field.
  10. Set the Write start address in the DMA4_CDSAi[31:0] DST_START_ADRS bit field.
  11. Set FS to 0 and BS to 1, respectively, in DMA4_CCRi[5] FS and DMA4_CCRi[18] BS.
  12. Set to 1 the channel enable bit DMA4_CCRi[7] EN.
- To configure an LCh to transfer one packet per DMA request:
  1. Set the number of DMA request associated to the current LCH in the DMA4_CCRi[20:19] SYNCHRO_CONTROL_UPPER and DMA4_CCRi[4:0] SYNCHRO bit field.
  2. Set the data type, also referenced as element size (ES), in the DMA4_CSDPi[1:0] DATA_TYPE bit field.
  3. Set the number of elements per packet to transfer: If the packet requestor is in the source, set DMA4_CCRi [24] SEL_SRC_DST_SYNC to 1 and set the packet element number in the DMA4_CSFIi register and set the addressing mode of source to constant addressing in DMA4_CCRi[13:12] SRC_AMODE bit field; else, if the packet requestor is in the destination, set the DMA4_CCRi[24] SEL_SRC_DST_SYNC to 0 and set the packet element number in the DMA4_CDFIi register and set the addressing mode of destination to constant addressing in DMA4_CCRi[15:14] DST_AMODE bit field.
  4. Set the number of elements per frame in the DMA4_CENi[23:0] CHANNEL_ELMNT_NBR bit field.
  5. Set in the DMA4_CFNi[15:0] CHANNEL_FRAME_NBR bit field the number of frames (transfers), to take place before the LCH gets disabled.
  6. Set the element number in the packet in the DMA4_CSFIi[15:0] PKT_ELNT_NBR, if constant addressing or post-incremented addressing modes are used in the source side. However, the number of element in the packet is set in the DMA4_CDFIi[15:0] PKT_ELNT_NBR if constant addressing mode is used in the destination side.
  7. Set the Read Port access type (single or burst access) in the DMA4_CSDPi[8:7] SRC_BURST_EN bit field.
  8. Set the Write Port access type (single or burst access) in the DMA4_CSDPi[15:14] DST_BURST_EN bit field.
  9. Set the Read Port addressing mode in the DMA4_CCRi[13:12] SRC_AMODE bit field.

10. Set the Write Port addressing mode in the DMA4_CCRi[15:14] DST_AMODE bit field.
11. Set the Read start address in the DMA4_CSSAi[31:0] SRC_START_ADRS bit field.
12. Set the Write start address in the DMA4_CDSAi[31:0] DST_START_ADRS bit field.
13. Set FS to 1 and BS to 1, respectively, in DMA4_CCRi[5] FS and DMA4_CCRi[18] BS.
14. Set to 1 the channel enable bit DMA4_CCRi[7] EN.

NOTE: It is possible to stop a transfer by disabling the channel by resetting the DMA4_CCRi[7] ENABLE bit.

16.1.5.4 Synchronized Transfer Monitoring Using CDAC

The DMA4_CDACi register is writable and non-initialized (value undefined). It can be initialized to monitor a transfer by applying the following programming model:

1. Write 0 in the DMA4_CDACi register.
2. Enable the channel.
3. If a time-out occurs, read DMA4_CDACi register.
4. If DMA4_CDACi != 0 (it is the value configured in DMA4_CDACi):

   This indicates that the corresponding transfer has started. The user can then rely on DMA4_CCENi and DMA4_CCFNi element and frame counters.

   Otherwise, if DMA4_CDACi = 0 (it is the value configured in the DMA4_CDACi):

   This indicates that the corresponding transfer did not start.

16.1.5.5 Concurrent Software and Hardware Synchronization

This section describes thread allocation only; it does not describe the entire transfer. Because synchronized transfers are latency critical, you must allocate a thread at least on the synchronized target side.

Even for multiple concurrent channels, thread reservation ensures that when a hardware DMA request arrives, the read/write scheduler finds available thread(s) to initiate a channel schedule and issue a read/write transaction.

Consider six concurrent channels:

- Channels 0, 1, 2, and 3 are dedicated to memory-memory transfer; they are software triggered and not synchronized.
- Channel 4 is dedicated to memory-peripheral transfer, hardware triggered, and synchronized on the write side.
- Channel 5 is dedicated to peripheral-memory transfer, hardware triggered, and synchronized on the read side.

To perform thread reservation:

1. Allow thread reservation for priority channel 4 and channel 5:

   Reserve one thread (Read ThreadID 0) on the read port and one thread (Write ThreadID 0) on the write port: set the DMA4_GCR[13:12] HI_THREAD_RESERVED bit field to 0x1.
2. Specify channel priority:

   Channel 4 is a write high priority channel: set DMA4_CCRi[26] WRITE_PRIORITY = 1.

   Channel 5 is a read high priority channel: set DMA4_CCRi[6] READ_PRIORITY = 1.

16.1.5.6 Chained Transfer

A chained DMA transfer can be programmed as follows:

1. Configure the transfer parameters for each logical DMA channel in the chain as in step 1 for either the synchronized or non-synchronized transfers described in Section 16.1.5.5, *Concurrent Software and Hardware Synchronization*.

2. For each channel in the chain, configure the DMA4_CLNK_CTRLi register as follows:
   - Next logical DMA channel number (for a looping chained transfer link last channel to first channel number), in the DMA4_CLNK_CTRLi[4:0] NEXTLCH_ID bit field.
   - Include the logical channel to the chain and enable link by setting the DMA4_CLNK_CTRLi[15] ENABLE_LNK bit.
   - For a non-looping chain, the last logical channel in the chain must have the DMA4_CLNK_CTRLi[15] ENABLE_LNK bit set to 0 to indicate the end of the chain.
3. Enable the transfer through the enable bit in the first logical channel DMA4_CCRi[7] ENABLE bit. All other channels in the chain must be disabled. Each channel is enabled automatically in turn when the previous logical channel transfer completes. A non-synchronized transfer starts immediately; a hardware-synchronized transfer starts when the DMA request line corresponding to the first DMA channel in the chain is asserted.

To stop a looping chained transfer, disable the DMA4_CLNK_CTRLi[15] ENABLE_LNK bit (by setting it to 0x0), of the final channel transfer.

In the RAM-to-RAM copy example, to copy in loop, it is possible to link channel 10 on itself. The following line can be added in the channel configuration:

```
/* g) Set link for loop */
DMA4_CLINK_CTRL_CH10 =
      0x0000800A;
```

16.1.5.7 90-Degree Clockwise Image Rotation

The 90-degree clockwise image rotation example described in Section 16.1.4.5, *Addressing Modes*, can be programmed as follows:

1. Configure the transfer parameters in the logical DMA channel registers:
   - DMA4_CSDPi:
     – Transfer ES = 32-bit (32 bpp), DMA4_CSDPi[1:0] DATA_TYPE bit field
     – Read and write port access types = maximum burst size supported by memory device, DMA4_CSDPi[8:7] SRC_BURST_EN and DMA4_CSDPi[15:14] DST_BURST_EN bit fields
     – Source and destination endianism, DMA4_CSDPi[21] SRC_ENDIAN and DMA4_CSDPi[19] DST_ENDIAN bits
     – Write mode = posted with last element nonposted, DMA4_CSDPi[17:16] WRITE_MODE bit field
     – Source and destination packed = Yes (although destination writes do not benefit because EI1), DMA4_CSDPi[6] SRC_PACKED and DMA4_CSDPi[13] DST_PACKED bits
   - DMA4_CENi: EN = 240
   - DMA4_CFNi: FN per transfer block = 160
   - DMA4_CSSAi: Source start address = 0x100000
   - DMA4_CDSAi: destination start address = 0x20013E
   - DMA4_CCRi:
     – Read and write port addressing modes = double-index addressing mode for both or post-increment addressing on source and double-index addressing on destination, DMA4_CCRi[13:12] SRC_AMODE and DMA4_CCRi[15:14] DST_AMODE bit fields
     – Low or high priority, DMA4_CCRi[6] READ_PRIORITY bit
     – DMA request number = 0 (for software-triggered transfer), DMA4_CCRi[4:0] SYNCHRO_CONTROL and DMA4_CCRi[20:19] SYNCHRO_CONTROL_UPPER bit fields
   - DMA4_CSEIi: Source EI = 1
   - DMA4_CSFIi: Source frame index = 1
   - DMA4_CDEIi: destination EI = 637
   - DMA4_CDFIi: destination frame index = 152967
2. Start the transfer through the enable bit in the channel DMA4_CCRi register.

The following parameters are used to perform this rotation from 0x80C00000 RAM address to 0x80F00000, with the same code used in Section 16.1.5.2, *Software-Triggered (Nonsynchronized) Transfer*:

```
/* Init. parameters */
DMA4->DataType = 0x2; //
      DMA4_CSDPi[1:0]
DMA4->ReadPortAccessType = 0x3; // DMA4_CSDPi[8:7]

DMA4->WritePortAccessType = 0x3; // DMA4_CSDPi[15:14]

DMA4->SourceEndiansim = 0; // DMA4_CSDPi[21]

DMA4->DestinationEndianism = 0; // DMA4_CSDPi[19]

DMA4->WriteMode = 0x2; // DMA4_CSDPi[17:16]
DMA4->SourcePacked
      = 0x1; // DMA4_CSDPi[6]
DMA4->DestinationPacked = 0x1; //
      DMA4_CSDPi[13]
DMA4->NumberOfElementPerFrame = 240; // DMA4_CENi

DMA4->NumberOfFramePerTransferBlock = 160; // DMA4_CFNi

DMA4->SourceStartAddress = 0x80C00000; // DMA4_CSSAi

DMA4->DestinationStartAddress = 0x80F00000; // DMA4_CDSAi

DMA4->SourceElementIndex = 1; // DMA4_CSEIi

DMA4->SourceFrameIndex = 1; // DMA4_CSFIi

DMA4->DestinationElementIndex = 637; // DMA4_CDEIi

DMA4->DestinationFrameIndex = -152967; // DMA4_CDFIi

DMA4->ReadPortAccessMode = 0x3; // DMA4_CCRi[13:12]

DMA4->WritePortAccessMode = 0x3; // DMA4_CCRi[15:14]

DMA4->ReadPriority = 0; // DMA4_CCRi[6]
DMA4->WritePriority =
      0; // DMA4_CCRi[23]
DMA4->ReadRequestNumber = 0; // DMA4_CCRi[4:0]

DMA4->WriteRequestNumber = 0; // DMA4_CCRi[20:19]
```

16.1.5.8 Graphic Operations

- Transparent copy:
  1. Set the DMA4_CCRi[17] TRANSPARENT_COPY_ENABLE bit to 1
  2. Set the DMA4_CCRi[16] CONST_FILL_ENABLE bit to 0
  3. Set the value of the key color in the DMA4_COLORi[15:0] COLOR_KEY bit field

  To perform this graphic operation, the following lines can be added to the example of Section 16.1.5.2, *Software-Triggered (Nonsynchronized) Transfer*.

```
DMA4_CCR_CH10 &= ~(0x1 << 16);
DMA4_CCR_CH10 |= 0x1 << 17;

DMA4_COLOR_CH10 = 0x00000003;
```

- Solid Color fill:
  1. Set the DMA4_CCRi[16] CONST_FILL_ENABLE bit to 1
  2. Set the DMA4_CCRi[17] TRANSPARENT_COPY_ENABLE bit to 0

3. Set the value of key the color in the DMA4_COLORi[15:0] SOLID_COLOR bit field

   To perform this graphic operation, the following lines can be added to the example of Section 16.1.5.2, *Software-Triggered (Nonsynchronized) Transfer*.

```
DMA4_CCR_CH10 &= ~(0x1 << 17);
DMA4_CCR_CH10 |= 0x1 << 16;

DMA4_COLOR_CH10 = 0x00000003;
```

16.1.5.9 Linked-List Programming Guidelines

- With the exception of the DMA4_CCRi[7] ENABLE bit and the DMA4_CDPi[7] PAUSE_LINK_LIST bit during a linked-list transfer (descriptor load phase or data load phase), avoid programming any register through the configuration port.
- Before enabling any linked-list transfer, ensure that all global registers and all registers in the descriptor are initialized. Some static channel registers (registers that are not updated by the descriptor to be loaded) must also be initialized correctly:
  - For type 2, the following registers must be initialized with consistent values:
    - All global registers
    - DMA4_CCRi
    - DMA4_CSDPi
    - DMA4_CLNK_CTRLi
  - For type 3, the following registers must be initialized with consistent values:
    - All global registers
    - DMA4_CCRi
    - DMA4_CSDPi
    - DMA4_CLNK_CTRLi
    - DMA4_CICRi
    - DMA4_CFNi
- In case of a linked list with descriptor types 2 and 3, the content of the DMA4_CCRi register must not change during super-block life.
- The PAUSE_LINK_LIST bit must not be set in the initialization phase.

16.1.6 DMA_SYSTEM Register Manual

16.1.6.1 DMA_SYSTEM Instance Summary

Table 16-19. DMA_SYSTEM Instance Summary

Module Name	Base Address	Size
DMA_SYSTEM	0x4A05 6000	4 KiB

16.1.6.2 DMA_SYSTEM Registers

16.1.6.2.1 DMA_SYSTEM Register Summary

Index *i* represents the logical channel number (where i = 0 to 31). The offset address for some registers is calculated from the channel *c* number. For example, the DMA4_CCR10 (channel 10) register has an offset address of 10 × 0x60 = 0x3C0, and thus a physical address of 0x4A05 6080 + 0x3C0 = 0x4A05 6440.

Index *j* represents the interrupt line number (where j = 0 to 3) The offset address for some registers is calculated from the channel *c* number. For example, the DMA4_IRQSTATUS_L3 (line 3) register has an offset address of 3 × 0x4 = 0xC, and thus a physical address of 0x4A05 6008 + 0xC = 0x4A05 6014.

Table 16-20. DMA_SYSTEM Registers Mapping Summary

Register Name	Type	Register Width (Bits)	Address Offset	DMA_SYSTEM Base Address
DMA4_REVISION	R	32	0x0000 0000	0x4A05 6000
DMA4_IRQSTATUS_Lj [(1)]	RW	32	0x0000 0008 + (0x4 * j)	0x4A05 6008 + (0x4 * j)
DMA4_IRQENABLE_Lj [(1)]	RW	32	0x0000 0018 + (0x4 * j)	0x4A05 6018 + (0x4 * j)
DMA4_SYSSTATUS	R	32	0x0000 0028	0x4A05 6028
DMA4_OCP_SYSCONFIG	RW	32	0x0000 002C	0x4A05 602C
DMA4_CAPS_0	RW	32	0x0000 0064	0x4A05 6064
DMA4_CAPS_2	R	32	0x0000 006C	0x4A05 606C
DMA4_CAPS_3	R	32	0x0000 0070	0x4A05 6070
DMA4_CAPS_4	RW	32	0x0000 0074	0x4A05 6074
DMA4_GCR	RW	32	0x0000 0078	0x4A05 6078
DMA4_CCRi [(2)]	RW	32	0x0000 0080 + (0x60 * i)	0x4A05 6080 + (0x60 * i)
DMA4_CLNK_CTRLi [(2)]	RW	32	0x0000 0084 + (0x60 * i)	0x4A05 6084 + (0x60 * i)
DMA4_CICRi [(2)]	RW	32	0x0000 0088 + (0x60 * i)	0x4A05 6088 + (0x60 * i)
DMA4_CSRi [(2)]	RW	32	0x0000 008C + (0x60 * i)	0x4A05 608C + (0x60 * i)
DMA4_CSDPi [(2)]	RW	32	0x0000 0090 + (0x60 * i)	0x4A05 6090 + (0x60 * i)
DMA4_CENi [(2)]	RW	32	0x0000 0094 + (0x60 * i)	0x4A05 6094 + (0x60 * i)
DMA4_CFNi [(2)]	RW	32	0x0000 0098 + (0x60 * i)	0x4A05 6098 + (0x60 * i)
DMA4_CSSAi [(2)]	RW	32	0x0000 009C + (0x60 * i)	0x4A05 609C + (0x60 * i)
DMA4_CDSAi [(2)]	RW	32	0x0000 00A0 + (0x60 * i)	0x4A05 60A0 + (0x60 * i)
DMA4_CSEIi [(2)]	RW	32	0x0000 00A4 + (0x60 * i)	0x4A05 60A4 + (0x60 * i)
DMA4_CSFIi [(2)]	RW	32	0x0000 00A8 + (0x60 * i)	0x4A05 60A8 + (0x60 * i)
DMA4_CDEIi [(2)]	RW	32	0x0000 00AC + (0x60 * i)	0x4A05 60AC + (0x60 * i)
DMA4_CDFIi [(2)]	RW	32	0x0000 00B0 + (0x60 * i)	0x4A05 60B0 + (0x60 * i)
DMA4_CSACi [(2)]	R	32	0x0000 00B4 + (0x60 * i)	0x4A05 60B4 + (0x60 * i)
DMA4_CDACi [(2)]	RW	32	0x0000 00B8 + (0x60 * i)	0x4A05 60B8 + (0x60 * i)
DMA4_CCENi [(2)]	RW	32	0x0000 00BC + (0x60 * i)	0x4A05 60BC + (0x60 * i)
DMA4_CCFNi [(2)]	RW	32	0x0000 00C0 + (0x60 * i)	0x4A05 60C0 + (0x60 * i)
DMA4_COLORi [(2)]	RW	32	0x0000 00C4 + (0x60 * i)	0x4A05 60C4 + (0x60 * i)

(1) j = 0 to 3
(2) i = 0 to 31

Table 16-20. DMA_SYSTEM Registers Mapping Summary (continued)

Register Name	Type	Register Width (Bits)	Address Offset	DMA_SYSTEM Base Address
DMA4_CDPi [(2)]	RW	32	0x0000 00D0 + (0x60 * i)	0x4A05 60D0 + (0x60 * i)
DMA4_CNDPi [(2)]	RW	32	0x0000 00D4 + (0x60 * i)	0x4A05 60D4 + (0x60 * i)
DMA4_CCDNi [(2)]	RW	32	0x0000 00D8 + (0x60 * i)	0x4A05 60D8 + (0x60 * i)

16.1.6.2.2 DMA_SYSTEM Register Description

NOTE: Some registers have no reset value (marked with -) because of hardware implementation in memory. Software must ensure the correct programming of these registers, if needed.

Shadow registers are used to read run-time registers such as CCEN, CCFN, CDAC, and CSAC. Typically, when accessed in 8-bit or 16-bit access for two consecutive accesses, the value of the previous registers can change. A shadow register holds the entire value to let the next access recover the remaining 24 or 16 bits.

For non-32-bit transactions, start reading or writing from the LSByte first to enable the register update. There is no issue for 32-bit read-write transactions.

Table 16-21. DMA4_REVISION

Address Offset	0x0000 0000		
Physical Address	0x4A05 6000	**Instance**	DMA_SYSTEM
Description	This register contains the DMA revision code		
Type	R		

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

Bits	Field Name	Description	Type	Reset
31:0	REVISION	Reserved, Write 0's for future compatibility. Read returns 0	R	TI internal Data

Table 16-22. Register Call Summary for Register DMA4_REVISION

DMA_SYSTEM Register Manual

- DMA_SYSTEM Register Summary: [0]

Table 16-23. DMA4_IRQSTATUS_Lj

Address Offset	0x0000 0008 + (0x4 * j)	**Index**	j = 0 to 3
Physical Address	0x4A05 6008 + (0x4 * j)	**Instance**	DMA_SYSTEM
Description	The interrupt status register regroups all the status of the DMA_SYSTEM channels that can generate an interrupt over line Lj.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH_31_0_Lj																															

Bits	Field Name	Description	Type	Reset
31:0	CH_31_0_Lj	Channel 31 Interrupt on Lj: When an interrupt is seen on the line Lj the status of a interrupting channel i is read in the bit field i. Read 0x0: Channel Interrupt Lj false Write 0x0: Channel Interrupt Lj status bit unchanged Write 0x1: Channel Interrupt Lj status bit is reset Read 0x1: Channel Interrupt Lj true (pending)	RW W1toClr	0x0000 0000

Table 16-24. Register Call Summary for Register DMA4_IRQSTATUS_Lj

DMA_SYSTEM Functional Description
- DMA_SYSTEM Controller Interrupt Requests: [0] [1] [2] [3]
- Interrupt Generation: [4]

DMA_SYSTEM Register Manual
- DMA_SYSTEM Register Summary: [5]

Table 16-25. DMA4_IRQENABLE_Lj

Address Offset	0x0000 0018 + (0x4 * j)	**Index**	j = 0 to 3
Physical Address	0x4A05 6018 + (0x4 * j)	**Instance**	DMA_SYSTEM
Description	The interrupt enable register allows to mask/unmask the module internal sources of interrupt, on line Lj		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH_31_0_Lj_EN																															

Bits	Field Name	Description	Type	Reset
31:0	CH_31_0_Lj_EN	Channel Interrupt on Lj mask/unmask : to Mask/Unmask a channel i interrupt on Lj the user writes 0/1 on the bit field i. 0x0: Channel Interrupt Lj is masked 0x1: Channel Interrupt Lj generates an interrupt when it occurs	RW	0x0000 0000

Table 16-26. Register Call Summary for Register DMA4_IRQENABLE_Lj

DMA_SYSTEM Functional Description
- DMA_SYSTEM Controller Interrupt Requests: [0] [1] [2] [3]
- Interrupt Generation: [4]

DMA_SYSTEM Basic Programming Model
- Setup Configuration: [5]

DMA_SYSTEM Register Manual
- DMA_SYSTEM Register Summary: [6]

Table 16-27. DMA4_SYSSTATUS

Address Offset	0x0000 0028		
Physical Address	0x4A05 6028	**Instance**	DMA_SYSTEM
Description	The register provides status information about the module excluding the interrupt status information (see interrupt status register)		
Type	R		

31–1	0
RESERVED	RESETDONE

Bits	Field Name	Description	Type	Reset
31:1	RESERVED	Reserved for module-specific status information	R	0x0000 0000
0	RESETDONE	Internal reset monitoring Read 0x0: Internal module reset is on-going Read 0x1: Reset completed	R	1

Table 16-28. Register Call Summary for Register DMA4_SYSSTATUS

DMA_SYSTEM Register Manual
- DMA_SYSTEM Register Summary: [0]

Table 16-29. DMA4_OCP_SYSCONFIG

Address Offset	0x0000 002C		
Physical Address	0x4A05 602C	**Instance**	DMA_SYSTEM
Description	DMA system configuration register		
Type	RW		

31–14	13–12	11–10	9–8	7–6	5	4–3	2	1	0
RESERVED	MIDLEMODE	RESERVED	CLOCKACTIVITY	RESERVED	EMUFREE	SIDLEMODE	RESERVED	RESERVED	AUTOIDLE

Bits	Field Name	Description	Type	Reset
31:14	RESERVED	Write 0's for future compatibility, Reads return 0	RW	0x00000
13:12	MIDLEMODE	Read write power management, standby/wait control 0x0: Force-standby: MStandby is asserted only when all the DMA channels are disabled 0x1: No-Standby: MStandby is never asserted 0x2: Smart-Standby: MStandby is asserted if at least one of the following two conditions is satisfied: 1. All the channels are disabled, OR 2. There is no non-synchronized channel enabled AND [if hardware synchronized channel is enabled, then no DMA request input is asserted and no requests are pending to be serviced]. 0x3: Reserved	RW	0x0
11:10	RESERVED	Reserved for clocks activities extension	RW	0x0

Bits	Field Name	Description	Type	Reset
9:8	CLOCKACTIVITY	Clocks activities during wake-up Bit 8: Interface clock 0x0: Interface clock can be switched-off Bit 9: Functional clock 0x0: Functional clock can be switched-off	R	0x0
7:6	RESERVED	Write 0's for future compatibility. Read returns 0	RW	0x0
5	EMUFREE	Enable sensitivity to MSuspend 0x0: DMA4 freezes its internal logic upon MSuspend assertion 0x1: DMA4 ignores the MSuspend input	RW	0
4:3	SIDLEMODE	Configuration port power management, Idle req/ack control 0x0: Force-idle. An idle request is acknowledged unconditionally 0x1: No-idle. An idle request is never acknowledged 0x2: Smart-idle. Idle acknowledge is given by DMA4 if all of the conditions are true: 1. All the channels are disabled. 2. If hardware synchronized channel is enabled, then no DMA request input is asserted and no requests are pending to be serviced. 3. All transactions are completed on all the DMA ports. 4.No interrupts are pending to be serviced. 0x3: Reserved. Do not use	RW	0x0
2	RESERVED	Write 0's for future compatibility, Reads return 0	RW	0
1	RESERVED	Reserved for non-GP devices	RW	0
0	AUTOIDLE	Internal interface clock gating strategy 0x0: Interface clock is free running 0x1: Automatic interface clock gating strategy is applied, based on the interface activity.	RW	0

Table 16-30. Register Call Summary for Register DMA4_OCP_SYSCONFIG

DMA_SYSTEM Functional Description
- DMA_SYSTEM Controller Power Management: [0] [1] [2] [3]

DMA_SYSTEM Register Manual
- DMA_SYSTEM Register Summary: [4]

Table 16-31. DMA4_CAPS_0

Address Offset	0x0000 0064		
Physical Address	0x4A05 6064	**Instance**	DMA_SYSTEM
Description	DMA Capabilities Register 0 LSW		
Type	RW		

31:22	21	20	19	18	17:0
RESERVED	LINK_LIST_CPBLTY_TYPE4	LINK_LIST_CPBLTY_TYPE123	CONST_FILL_CPBLTY	TRANSPARENT_BLT_CPBLTY	RESERVED

Bits	Field Name	Description	Type	Reset
31:22	RESERVED	Write 0's for future compatibility. Read returns 0	RW	0x000
21	LINK_LIST_CPBLTY_TYPE4	Link List capability for type4 descriptor capability	R	0
20	LINK_LIST_CPBLTY_TYPE123	Link List capability for type123 descriptor capability	R	1
19	CONST_FILL_CPBLTY	Constant_Fill_Capability Read 0x0: No LCH supports constant fill copy Read 0x1: any LCH supports constant fill copy	R	1
18	TRANSPARENT_BLT_CPBLTY	Transparent_BLT_Capability Read 0x0: No LCH supports transparent BLT copy Read 0x1: any LCH supports transparent BLT copy	R	1
17:0	RESERVED	Write 0's for future compatibility. Read returns 0	RW	0x00000

Table 16-32. Register Call Summary for Register DMA4_CAPS_0

DMA_SYSTEM Register Manual

- DMA_SYSTEM Register Summary: [0]

Table 16-33. DMA4_CAPS_2

Address Offset	0x0000 006C		
Physical Address	0x4A05 606C	**Instance**	DMA_SYSTEM
Description	DMA Capabilities Register 2		
Type	R		

31:9	8	7	6	5	4	3	2	1	0
RESERVED	SEPARATE_SRC_AND_DST_INDEX_CPBLTY	DST_DOUBLE_INDEX_ADRS_CPBLTY	DST_SINGLE_INDEX_ADRS_CPBLTY	DST_POST_INCRMNT_ADRS_CPBLTY	DST_CONST_ADRS_CPBLTY	SRC_DOUBLE_INDEX_ADRS_CPBLTY	SRC_SINGLE_INDEX_ADRS_CPBLTY	SRC_POST_INCREMENT_ADRS_CPBLTY	SRC_CONST_ADRS_CPBLTY

Bits	Field Name	Description	Type	Reset
31:9	RESERVED	Write 0's for future compatibility. Read returns 0	R	0x000000
8	SEPARATE_SRC_AND_DST_INDEX_CPBLTY	Separate_source/destination_index_capability Read 0x0: Does not support separate src/dst index for 2D addressing Read 0x1: Supports separate src/dest index for 2D addressing	R	1
7	DST_DOUBLE_INDEX_ADRS_CPBLTY	Destination_double_index_address_capability Read 0x0: Does not support double index address mode on the destination port Read 0x1: Supports double index address mode on the destination port	R	1
6	DST_SINGLE_INDEX_ADRS_CPBLTY	Destination_single_index_address_capability Read 0x0: Does not support single index address mode on the destination port Read 0x1: Supports single index address mode on the destination port	R	1
5	DST_POST_INCRMNT_ADRS_CPBLTY	Destination_post_increment_address_capability Read 0x0: Does not supports post-increment address mode in the destination port Read 0x1: Supports post-increment address mode in the destination port	R	1
4	DST_CONST_ADRS_CPBLTY	Destination_constant_address_capability Read 0x0: Does not supports constant address mode in the destination port Read 0x1: Supports constant address mode in the destination port	R	1

Bits	Field Name	Description	Type	Reset
3	SRC_DOUBLE_INDEX_ADRS_CPBLTY	Source_double_index_address_capability Read 0x0: Does not support double index address mode on the source port Read 0x1: Supports double index address mode on the source port	R	1
2	SRC_SINGLE_INDEX_ADRS_CPBLTY	Source_single_index_address_capability Read 0x0: Does not support single index address mode on the source port Read 0x1: Supports single index address mode in the source port	R	1
1	SRC_POST_INCREMENT_ADRS_CPBLTY	Source_post_increment_address_capability Read 0x0: Does not supports post-increment address mode in the source port Read 0x1: Supports post-increment address mode in the source port	R	1
0	SRC_CONST_ADRS_CPBLTY	Source_constant_address_capability Read 0x0: Does not supports constant address mode in the source port Read 0x1: Supports constant address mode in the source port	R	1

Table 16-34. Register Call Summary for Register DMA4_CAPS_2

DMA_SYSTEM Register Manual

- DMA_SYSTEM Register Summary: [0]

Table 16-35. DMA4_CAPS_3

Address Offset	0x0000 0070		
Physical Address	0x4A05 6070	**Instance**	DMA_SYSTEM
Description	DMA Capabilities Register 3		
Type	R		

31–8	7	6	5	4	3–2	1	0
RESERVED	BLOCK_SYNCHR_CPBLTY	PKT_SYNCHR_CPBLTY	CHANNEL_CHANINIG_CPBLTY	CHANNEL_INTERLEAVE_CPBLTY	RESERVED	FRAME_SYNCHR_CPBLTY	ELMNT_SYNCHR_CPBLTY

Bits	Field Name	Description	Type	Reset
31:8	RESERVED	Write 0's for future compatibility. Read returns 0	R	0x000000
7	BLOCK_SYNCHR_CPBLTY	Block_synchronization_capability Read 0x0: Does not support synchronization transfer on block boundary Read 0x1: Supports synchronization transfer on block boundary	R	1

Bits	Field Name	Description	Type	Reset
6	PKT_SYNCHR_CPBLTY	Packet_synchronization_capability Read 0x0: Does not support synchronization transfer on packet boundary Read 0x1: Supports synchronization transfer on packet boundary	R	1
5	CHANNEL_CHANINIG_CPBLTY	Channel_Chaninig_capability Read 0x0: Does not support Channel Chaninig capability Read 0x1: Supports Channel Chaninig capability	R	1
4	CHANNEL_INTERLEAVE_CPBLTY	Channel_interleave_capability Read 0x0: Does not support Channel interleave capability Read 0x1: Supports Channel_interleave capability	R	1
3:2	RESERVED		R	0x0
1	FRAME_SYNCHR_CPBLTY	Frame_synchronization_capability Read 0x0: Does not support synchronization transfer on Frame boundary Read 0x1: Supports synchronization transfer on Frame boundary	R	1
0	ELMNT_SYNCHR_CPBLTY	Element_synchronization_capability Read 0x0: Does not support synchronization transfer on Element boundary Read 0x1: Supports synchronization transfer on Element boundary	R	1

Table 16-36. Register Call Summary for Register DMA4_CAPS_3

DMA_SYSTEM Register Manual

- DMA_SYSTEM Register Summary: [0]

Table 16-37. DMA4_CAPS_4

Address Offset	0x0000 0074		
Physical Address	0x4A05 6074	**Instance**	DMA_SYSTEM
Description	DMA Capabilities Register 4		
Type	RW		

Bits	Field
31:15	RESERVED
14	EOSB_INTERRUPT_CPBLTY
13	RESERVED
12	DRAIN_END_INTERRUPT_CPBLTY
11	MISALIGNED_ADRS_ERR_INTERRUPT_CPBLTY
10	SUPERVISOR_ERR_INTERRUPT_CPBLTY
9	RESERVED
8	TRANS_ERR_INTERRUPT_CPBLTY
7	PKT_INTERRUPT_CPBLTY
6	SYNC_STATUS_CPBLTY
5	BLOCK_INTERRUPT_CPBLTY
4	LAST_FRAME_INTERRUPT_CPBLTY
3	FRAME_INTERRUPT_CPBLTY
2	HALF_FRAME_INTERRUPT_CPBLTY
1	EVENT_DROP_INTERRUPT_CPBLTY
0	RESERVED

Bits	Field Name	Description	Type	Reset
31:15	RESERVED	Write 0's for future compatibility. Read returns 0.	RW	0x00000
14	EOSB_INTERRUPT_CPBLTY	End of Super Block detection capability.	R	1
13	RESERVED	Reserved for non-GP devices	R	1
12	DRAIN_END_INTERRUPT_CPBLTY	Drain End detection capability.	R	1
11	MISALIGNED_ADRS_ERR_INTERRUPT_CPBLTY	Misaligned error detection capability.	R	1
10	SUPERVISOR_ERR_INTERRUPT_CPBLTY	Supervisor error detection capability.	R	1
9	RESERVED	Reserved for non-GP devices	R	1
8	TRANS_ERR_INTERRUPT_CPBLTY	Transaction error detection capability.	R	1
7	PKT_INTERRUPT_CPBLTY	End of Packet detection capability. Read 0x0: Does not support end of packet interrupt generation capability Read 0x1: Supports end of packet interrupt generation capability	R	1
6	SYNC_STATUS_CPBLTY	Sync_status_capability Read 0x0: Does not support synchronized transfer status bit generation Read 0x1: Supports synchronized transfer status bit generation	R	1
5	BLOCK_INTERRUPT_CPBLTY	End of block detection capability. Read 0x0: Does not support end of block interrupt generation capability Read 0x1: Supports end of block interrupt generation capability	R	1
4	LAST_FRAME_INTERRUPT_CPBLTY	Start of last frame detection capability. Read 0x0: Does not support last frame interrupt generation capability Read 0x1: Supports last frame interrupt generation capability	R	1
3	FRAME_INTERRUPT_CPBLTY	End of frame detection capability. Read 0x0: Does not support end of frame interrupt generation capability Read 0x1: Supports end of frame interrupt generation capability	R	1
2	HALF_FRAME_INTERRUPT_CPBLTY	Detection capability of the half of frame end. Read 0x0: Does not support half of frame interrupt generation capability Read 0x1: Supports half of frame interrupt generation capability	R	1
1	EVENT_DROP_INTERRUPT_CPBLTY	Request collision detection capability. Read 0x0: Does not support event drop interrupt generation capability Read 0x1: Supports event drop interrupt generation capability	R	1
0	RESERVED	Write 0's for future compatibility. Read returns 0.	RW	0

Table 16-38. Register Call Summary for Register DMA4_CAPS_4

DMA_SYSTEM Register Manual

- DMA_SYSTEM Register Summary: [0]

Table 16-39. DMA4_GCR

Address Offset	0x0000 0078		
Physical Address	0x4A05 6078	**Instance**	DMA_SYSTEM
Description	FIFO sharing between high and low priority channel. The Maximum per channel FIFO depth is bounded by the low and high channel FIFO budget. The high respectively low priority channels maximum burst size must be less than the min (high respectively low priority channel FIFO budget, per channel maximum FIFO depth)		
Type	RW		

31:25	24	23:16	15:14	13:12	11:8	7:0
RESERVED	CHANNEL_ID_GATE	ARBITRATION_RATE	HI_LO_FIFO_BUDGET	HI_THREAD_RESERVED	RESERVED	MAX_CHANNEL_FIFO_DEPTH

Bits	Field Name	Description	Type	Reset
31:25	RESERVED	Write 0's for future compatibility. Read returns 0.	RW	0x00
24	CHANNEL_ID_GATE	Gates the Channel ID bus monitoring on both Read and Write ports 0x0: Gates the Channel ID qualifiers on both Read and Write Ports 0x1: Does not gate the Channel ID qualifiers on both Read and Write Ports	RW	0x0
23:16	ARBITRATION_RATE	Arbitration switching rate between prioritized and regular channel queues	RW	0x01
15:14	HI_LO_FIFO_BUDGET	Allow to have a separate Global FIFO budget for high and low priority channels. For Hi priority Channel: (Per_channel_Maximum FIFO depth + 1) x Number of active High priority Channel =< High Budget FIFO For Low priority channel: (Per_channel_Maximum FIFO depth + 1) x Number of active Low priority Channel =< Low Budget FIFO 0x0: no fixed budget for neither higher nor lower priority channel 0x1: 75% of FIFO for low priority and 25% for high priority channels 0x2: 25% of FIFO for low priority and 75% for high priority channels 0x3: 50% of FIFO for low priority and 50% for high priority channels	RW	0x0
13:12	HI_THREAD_RESERVED	Allow thread reservation for high priority channel on both read and write ports. 0x0: No ThreadID is reserved on the Read Port for high priority channels. No ThreadID is reserved on the Write Port for high priority channels. 0x1: Read Port ThreadID 0 is reserved for high priority channels. Write Port ThreadID 0 is reserved for high priority channels. 0x2: Read port ThreadID 0 and ThreadID 1 are reserved for high priority channels. Write Port ThreadID 0 is reserved for high priority channels. 0x3: Read PortThreadID 0, ThreadID 1 and ThreadID 2 are reserved for high priority channels. Write Port ThreadID 0 is reserved for high priority channels.	RW	0x0

Bits	Field Name	Description	Type	Reset
11:8	RESERVED	Write 0's for future compatibility. Read returns 0.	RW	0x0
7:0	MAX_CHANNEL_FIFO_DEPTH	Maximum FIFO depth allocated to one logical channel. Maximum FIFO depth can not be 0x0. It should be at least 0x1 or greater. Note that If channel limit is less than destination burst size enough data will not be accumulated in the data FIFO and it will never be sent out on the WR port. The burst size should be less than the FIFO limit specified in this bit field.	RW	0x10

Table 16-40. Register Call Summary for Register DMA4_GCR

DMA_SYSTEM Functional Description
- Logical Channel Transfer Overview: [0] [1] [2]
- FIFO Queue Memory Pool: [3]
- Thread Budget Allocation: [4] [5] [6] [7] [8] [9] [10] [11] [12] [13] [14]
- FIFO Budget Allocation: [15] [16]

DMA_SYSTEM Basic Programming Model
- Setup Configuration: [17]
- Concurrent Software and Hardware Synchronization: [18]

DMA_SYSTEM Register Manual
- DMA_SYSTEM Register Summary: [19]

Table 16-41. DMA4_CCRi

Address Offset	0x0000 0080 + (0x60 * i)	**index:**	i = 0 to 31
Physical Address	0x4A05 6080 + (0x60 * i)	**Instance**	DMA_SYSTEM
Description	Channel Control Register		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED		RESERVED			WRITE_PRIORITY	BUFFERING_DISABLE	SEL_SRC_DST_SYNC	PREFETCH	SUPERVISOR	RESERVED	SYNCHRO_CONTROL_UPPER		BS	TRANSPARENT_COPY_ENABLE	CONST_FILL_ENABLE	DST_AMODE		SRC_AMODE		RESERVED	WR_ACTIVE	RD_ACTIVE	SUSPEND_SENSITIVE	ENABLE	READ_PRIORITY	FS	SYNCHRO_CONTROL				

Bits	Field Name	Description	Type	Reset
31:30	RESERVED	Write 0's for future compatibility. Read returns 0.	RW	0x0
29:27	RESERVED	Reserved for non-GP devices	RW	0x0
26	WRITE_PRIORITY	Channel priority on the Write side 0x0: Channel has low priority on the write side during the arbitration process. 0x1: Channel has high priority on write sided during the arbitration process.	RW	0

Bits	Field Name	Description	Type	Reset
25	BUFFERING_DISABLE	This bit allows to disable the default buffering functionality when transfer is source synchronized. 0x0: Buffering is enabled across element/packet when source is synchronized to element, packet, frame or blocks. 0x1: Buffering is disabled across element/packet when source is synchronized to element, packet, frame or blocks.	RW	-
24	SEL_SRC_DST_SYNC	Specifies that element, packet, frame or block transfer (depending on CCR.bs and CCR.fs) is triggered by the source or the destination on the DMA request 0x0: Transfer is triggered by the destination. If synch on packet the packet element number is specified in the CDFI register. 0x1: Transfer is triggered by the source. If synchronized on packet the packet element number is specified in the CSFI register.	RW	-
23	PREFETCH	Enables the prefetch mode 0x0: Prefetch mode is disabled. When Sel_Src_Dst_Sync=1 transfers are buffered and pipelined between DMA requests. 0x1: Prefetch mode is enabled. Prefetch mode is active only when destination is synchronized. It is software user responsibility not to have at the same time Prefetch=1 when Sel_Src_Dst_Sync=1. This mode is not supported.	RW	0
22	SUPERVISOR	Enables the supervisor mode 0x0: Supervisor mode is disabled. 0x1: Supervisor mode is enabled.	RW	0
21	RESERVED	Reserved for non-GP devices	RW	0
20:19	SYNCHRO_CONTROL_UPPER	Channel Synchronization control upper (used in conjunction with the 5 bits of synchro channel DMA4_CCRi[4:0]) Used in conjunction, as 2 MSB, with the 5 bits of the synchro channel bit field.	RW	0b00
18	BS	Block synchronization This bit used in conjunction with the fs to see how the DMA request is serviced in a synchronized transfer.	RW	-
17	TRANSPARENT_COPY_ENABLE	Transparent copy enable 0x0: Transparent copy mode is disabled. 0x1: Transparent copy mode is enabled.	RW	-
16	CONST_FILL_ENABLE	Constant fill enable 0x0: Constant fill mode is disabled. 0x1: Constant fill mode is enabled.	RW	0
15:14	DST_AMODE	Selects the addressing mode on the Write Port of a channel. 0x0: Constant address mode 0x1: Post-incremented address mode 0x2: Single index address mode 0x3: Double index address mode	RW	0bxx
13:12	SRC_AMODE	Selects the addressing mode on the Read Port of a channel. 0x0: Constant address mode 0x1: Post-incremented address mode 0x2: Single index address mode 0x3: Double index address mode	RW	0bxx
11	RESERVED	Write 0s for future compatibility. Read returns 0.	RW	0

Bits	Field Name	Description	Type	Reset
10	WR_ACTIVE	Indicates if the channel write context is active or not Read 0x0: Channel is not active on the write port. Read 0x1: Channel is currently active on the write port.	R	0
9	RD_ACTIVE	Indicates if the channel read context is active or not Read 0x0: Channel is not active on the read port. Read 0x1: Channel is currently active on the read port.	R	0
8	SUSPEND_SENSITIVE	Logical channel suspend enable bit 0x0: The channel ignores the MSuspend even if EMUFree is set to 0. 0x1: If EMUFree is set to 0 and MSuspend comes in then all current OCP services (single transaction or burst transaction as specified in the corresponding CSDP register) have to be completed before stopping processing any more transactions.	RW	0
7	ENABLE	Logical channel enable. It is SW responsibility to clear the CSR register and the IRQSTATUS bit for the different interrupt lines before enabling the channel. 0x0: The logical channel is disabled. 0x1: The logical channel is enabled.	RW	0
6	READ_PRIORITY	Channel priority on the read side 0x0: Channel has low priority on the read side during the arbitration process. 0x1: Channel has high priority on read sided during the arbitration process.	RW	0
5	FS	Frame synchronization This bit used in conjunction with the BS to see how the DMA request is serviced in a synchronized transfer FS = 0 and BS = 0: An element is transferred once a DMA request is made. FS = 0 and BS = 1: An entire block is transferred once a DMA request is made. FS = 1 and BS = 0: An entire frame is transferred once a DMA request is made. FS = 1 and BS = 1: A packet is transferred once a DMA request is made. All these different transfers can be interleaved on the port with other DMA requests.	RW	-
4:0	SYNCHRO_CONTROL	Channel synchronization control This bit field used in conjunction with the second_level_ synchro_control_upper (as 2 MSB) 0000000 : Is reserved for non synchronized LCH transfer xxxxxxx (from 1 to 127)There are 127 possible DMA request to assign to any LCH. **Note:** The channel synchronization control registers are 1-based. For example, to enable the S_DMA_1 request, DMA4_CCR[4:0] SYNCHRO_CONTROL must be set to 0x2 (DMA request number + 1).	RW	0b00000

Table 16-42. Register Call Summary for Register DMA4_CCRi

DMA_SYSTEM Functional Description
- Interrupt Generation: [0]
- Logical Channel Transfer Overview: [1] [2] [3]
- Addressing Modes: [4]
- Software Synchronization: [5] [6] [7]
- Hardware Synchronization: [8] [9] [10] [11] [12] [13] [14] [15] [16] [17] [18] [19]
- Thread Budget Allocation: [20] [21]
- Reprogramming an Active Channel: [22] [23] [24] [25]
- Packet Synchronization: [26] [27] [28] [29] [30] [31]
- Graphics Acceleration Support: [32]
- Supervisor Modes: [33]
- Disabling a Channel During Transfer: [34]
- FIFO Draining Mechanism: [35] [36] [37] [38] [39]
- Descriptors: [40] [41]
- Linked-List Control and Monitoring: [42] [43] [44] [45] [46] [47] [48] [49] [50] [51]

DMA_SYSTEM Basic Programming Model
- Software-Triggered (Nonsynchronized) Transfer: [52] [53] [54] [55] [56] [57] [58] [59] [60]
- Hardware-Synchronized Transfer: [61] [62] [63] [64] [65] [66] [67] [68] [69] [70] [71] [72] [73] [74] [75] [76] [77] [78] [79] [80] [81] [82] [83] [84] [85] [86] [87] [88] [89] [90] [91] [92] [93] [94] [95] [96] [97] [98] [99]
- Concurrent Software and Hardware Synchronization: [100] [101]
- Chained Transfer: [102]
- 90-Degree Clockwise Image Rotation: [103] [104] [105] [106] [107] [108] [109]
- Graphic Operations: [110] [111] [112] [113]
- Linked-List Programming Guidelines: [114] [115] [116] [117]

DMA_SYSTEM Register Manual
- DMA_SYSTEM Register Summary: [118]
- DMA_SYSTEM Register Description: [119]

Table 16-43. DMA4_CLNK_CTRLi

Address Offset	0x0000 0084 + (0x60 * i)	**index:**	i = 0 to 31
Physical Address	0x4A05 6084 + (0x60 * i)	**Instance**	DMA_SYSTEM
Description	Channel Link Control Register		
Type	RW		

31–16	15	14–5	4–0
RESERVED	ENABLE_LNK	RESERVED	NEXTLCH_ID

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Write 0's for future compatibility. Read returns 0.	RW	0x0000
15	ENABLE_LNK	Enables or disable the channel linking. 0x0: Channel linking mode is disabled When set on the fly to 0 the current channel will complete the transfer and stops the chain linking 0x1: Channel linking mode is enabled. The logical channel defined in the NextLCH_ID is enabled at the end of the current transfer	RW	0
14:5	RESERVED	Write 0's for future compatibility. Read returns 0.	RW	0x000
4:0	NEXTLCH_ID	Defines the NextLCh_ID, which is used to build logical channel chaining queue.	RW	0bxxxxx

Table 16-44. Register Call Summary for Register DMA4_CLNK_CTRLi

DMA_SYSTEM Functional Description

- Chained Logical Channel Transfers: [0] [1]
- FIFO Draining Mechanism: [2]
- Descriptors: [3]
- Linked-List Control and Monitoring: [4] [5] [6] [7]

DMA_SYSTEM Basic Programming Model

- Chained Transfer: [8] [9] [10] [11] [12]
- Linked-List Programming Guidelines: [13] [14]

DMA_SYSTEM Register Manual

- DMA_SYSTEM Register Summary: [15]

Table 16-45. DMA4_CICRi

Address Offset	0x0000 0088 + (0x60 * i)	**index:**	i = 0 to 31
Physical Address	0x4A05 6088 + (0x60 * i)	**Instance**	DMA_SYSTEM
Description	Channel Interrupt Control Register		
Type	RW		

31:15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED	SUPER_BLOCK_IE	RESERVED	DRAIN_IE	MISALIGNED_ERR_IE	SUPERVISOR_ERR_IE	RESERVED	TRANS_ERR_IE	PKT_IE	RESERVED	BLOCK_IE	LAST_IE	FRAME_IE	HALF_IE	DROP_IE	RESERVED

Bits	Field Name	Description	Type	Reset
31:15	RESERVED	Write 0's for future compatibility. Read returns 0.	RW	0x00000
14	SUPER_BLOCK_IE	Enables the end of super block interrupt	RW	-
13	RESERVED	Reserved for non-GP devices	RW	1
12	DRAIN_IE	Enables the end of draining interrupt	RW	0
11	MISALIGNED_ERR_IE	Enables the address misaligned error event interrupt 0x0: Disables the misaligned address error event interrupt 0x1: Enables the misaligned address error event interrupt	RW	-
10	SUPERVISOR_ERR_IE	Enables the supervisor transaction error event interrupt 0x0: Disables the supervisor transaction error event interrupt 0x1: Enables the supervisor transaction error event interrupt	RW	1
9	RESERVED	Reserved for non-GP devices	RW	1
8	TRANS_ERR_IE	Enables the transaction error event interrupt 0x0: Disables the transaction error event interrupt 0x1: Enables the transaction error event interrupt	RW	-
7	PKT_IE	Enables the end of Packet interrupt 0x0: Disables the end of Packet transfer interrupt 0x1: Enables the end of Packet transfer interrupt	RW	-
6	RESERVED	Write 0's for future compatibility. Read returns 0.	RW	0

Bits	Field Name	Description	Type	Reset
5	BLOCK_IE	Enables the end of block interrupt 0x0: Disables the end of block interrupt 0x1: Disables the end of block interrupt	RW	-
4	LAST_IE	Last frame interrupt enable (start of last frame) 0x0: Disables the last frame interrupt 0x1: Enables the last frame interrupt	RW	-
3	FRAME_IE	Frame interrupt enable (end of frame) 0x0: Disables the end of frame interrupt 0x1: Enables the end of frame interrupt	RW	-
2	HALF_IE	Enables or disables the half frame interrupt. 0x0: Disables the half frame interrupt 0x1: Enables the half frame interrupt	RW	-
1	DROP_IE	Synchronization event drop interrupt enable (request collision) 0x0: Disables the event drop interrupt 0x1: Enables the event drop interrupt	RW	0
0	RESERVED	Write 0's for future compatibility. Read returns 0.	RW	0

Table 16-46. Register Call Summary for Register DMA4_CICRi

DMA_SYSTEM Functional Description
- DMA_SYSTEM Controller Interrupt Requests: [0] [1]
- Interrupt Generation: [2]
- FIFO Draining Mechanism: [3]
- Descriptors: [4] [5] [6]
- Linked-List Control and Monitoring: [7] [8] [9]

DMA_SYSTEM Basic Programming Model
- Setup Configuration: [10]
- Linked-List Programming Guidelines: [11]

DMA_SYSTEM Register Manual
- DMA_SYSTEM Register Summary: [12]

Table 16-47. DMA4_CSRi

Address Offset	0x0000 008C + (0x60 * i)	**index:**	i = 0 to 31
Physical Address	0x4A05 608C + (0x60 * i)	**Instance**	DMA_SYSTEM
Description	Channel Status Register		
Type	RW		

Bits	Field
31-17	RESERVED
16-15	RESERVED
14	SUPER_BLOCK
13	RESERVED
12	DRAIN_END
11	MISALIGNED_ADRS_ERR
10	SUPERVISOR_ERR
9	RESERVED
8	TRANS_ERR
7	PKT
6	SYNC
5	BLOCK
4	LAST
3	FRAME
2	HALF
1	DROP
0	RESERVED

Bits	Field Name	Description	Type	Reset
31:17	RESERVED	Write 0's for future compatibility. Read returns 0.	RW	0x0000
16:15	RESERVED	Reserved for debug (Monitor descriptor/data load phase), Write 0's for future compatibility, Read returns 0	RW	0x0
14	SUPER_BLOCK	End of Super block event Read 0x0: The current Super block transfer has not been finished Write 0x0: Status bit unchanged Read 0x1: The current Super block has been transferred Write 0x1: Status bit is reset	RW W1toClr	0
13	RESERVED	Reserved for non-GP devices	RW	0
12	DRAIN_END	End of channel draining Read 0x0: No drain end in the current transfer Write 0x0: Status bit unchanged Read 0x1: The current channel draining is completed Write 0x1: Status bit is reset	RW W1toClr	0
11	MISALIGNED_ADRS_ERR	Misaligned address error event Read 0x0: No address error Write 0x0: Status bit unchanged Write 0x1: Status bit is reset Read 0x1: An address error has been occurred	RW W1toClr	0
10	SUPERVISOR_ERR	Supervisor transaction error event Read 0x0: No supervisor transaction error Write 0x0: Status bit unchanged Write 0x1: Status bit is reset Read 0x1: A supervisor transaction error has been occurred	RW W1toClr	0
9	RESERVED	Reserved for non-GP devices	RW	0
8	TRANS_ERR	Transaction error event Read 0x0: No transaction error Write 0x0: Status bit unchanged Write 0x1: Status bit is reset Read 0x1: A transaction error has been occurred	RW W1toClr	0
7	PKT	End of Packet transfer Read 0x0: The current packet transfer has not been finished Write 0x0: Status bit unchanged Write 0x1: Status bit is reset Read 0x1: The current packet has been transferred	RW W1toClr	0
6	SYNC	Synchronization status of a channel. Read 0x0: Logical channel is not scheduled or servicing a non synchronized DMA request. Write 0x0: Status bit unchanged Write 0x1: Status bit is reset Read 0x1: Logical channel is servicing a synchronized DMA request	RW W1toClr	0

Bits	Field Name	Description	Type	Reset
5	BLOCK	End of block event Read 0x0: The current block transfer has not been finished Write 0x0: Status bit unchanged Write 0x1: Status bit is reset Read 0x1: The current block has been transferred	RW W1toClr	0
4	LAST	Last frame (start of last frame) Read 0x0: The start of the last frame to transfer is not reached Write 0x0: Status bit unchanged Write 0x1: Status bit is reset Read 0x1: The start of the last frame to transfer is reached	RW W1toClr	0
3	FRAME	End of frame event Read 0x0: The end of current transferred frame is not reached Write 0x0: Status bit unchanged Write 0x1: Status bit is reset Read 0x1: The end of current transferred frame is reached	RW W1toClr	0
2	HALF	Half of frame event. Read 0x0: The half of current transferred frame is not reached Write 0x0: Status bit unchanged Write 0x1: Status bit is reset Read 0x1: The half of current transferred frame is reached	RW W1toClr	0
1	DROP	Synchronization event drop occured during the transfer Read 0x0: No synchronization collision Write 0x0: Status bit unchanged Write 0x1: Status bit is reset Read 0x1: A synchronization collision has been occurred	RW W1toClr	0
0	RESERVED	Write 0's for future compatibility. Read returns 0.	RW	0

Table 16-48. Register Call Summary for Register DMA4_CSRi

DMA_SYSTEM Functional Description

- DMA_SYSTEM Controller Interrupt Requests: [0] [1] [2] [3] [4]
- Interrupt Generation: [5]
- FIFO Draining Mechanism: [6]
- Linked-List Control and Monitoring: [7] [8] [9] [10] [11]

DMA_SYSTEM Basic Programming Model

- Setup Configuration: [12]

DMA_SYSTEM Register Manual

- DMA_SYSTEM Register Summary: [13]

Table 16-49. DMA4_CSDPi

Address Offset	0x0000 0090 + (0x60 * i)	**index:**	i = 0 to 31
Physical Address	0x4A05 6090 + (0x60 * i)	**Instance**	DMA_SYSTEM
Description	Channel Source Destination Parameters		
Type	RW		

31:22	21	20	19	18	17:16	15:14	13	12:9	8	7	6	5:2	1:0
RESERVED	SRC_ENDIAN	SRC_ENDIAN_LOCK	DST_ENDIAN	DST_ENDIAN_LOCK	WRITE_MODE	DST_BURST_EN	DST_PACKED	RESERVED	SRC_BURST_EN		SRC_PACKED	RESERVED	DATA_TYPE

Bits	Field Name	Description	Type	Reset
31:22	RESERVED	Write 0's for future compatibility. Read returns 0.	RW	0x000
21	SRC_ENDIAN	Channel source endianness control 0x0: Source has Little Endian type 0x1: Source has Big Endian type	RW	-
20	SRC_ENDIAN_LOCK	Endianness Lock 0x0: Endianness adapt 0x1: Endianness lock	RW	-
19	DST_ENDIAN	Channel Destination endianness control 0x0: Destination has Little Endian type 0x1: Destination has Big Endian type	RW	-
18	DST_ENDIAN_LOCK	Endianness Lock 0x0: Endianness adapt 0x1: Endianness lock	RW	-
17:16	WRITE_MODE	Used to enable writing mode without posting or with posting 0x0: Write None Posted (WRNP) 0x1: Write (Posted) 0x2: All transaction are mapped on the Write command as posted except for the last transaction in the transfer mapped on a Write None Posted 0x3: Undefined	RW	0bxx
15:14	DST_BURST_EN	Used to enable bursting on the Write Port. Smaller burst size than the programmed burst size is also allowed 0x0: single access 0x1: 16 bytes or 4x32-bit / 2x64-bit burst access 0x2: 32 bytes or 8x32-bit / 4x64-bit burst access 0x3: 64 bytes or 16x32-bit / 8x64-bit burst access	RW	0b00
13	DST_PACKED	Destination receives packed data. 0x0: The destination target is non packed 0x1: The destination target is packed	RW	-
12:9	RESERVED	Write the reset value. Read returns reset value	RW	0x-

Bits	Field Name	Description	Type	Reset
8:7	SRC_BURST_EN	Used to enable bursting on the Read Port. Smaller burst size than the programmed burst size is also allowed 0x0: single access 0x1: 16 bytes or 4x32-bit / 2x64-bit burst access 0x2: 32 bytes or 8x32-bit / 4x64-bit burst access 0x3: 64 bytes or 16x32-bit / 8x64-bit burst access	RW	0bxx
6	SRC_PACKED	Source provides packed data. 0x0: The source target is non packed 0x1: The source target is packed	RW	-
5:2	RESERVED	Write the reset value. Read returns reset value	RW	0x-
1:0	DATA_TYPE	Defines the type of the data moved in the channel. 0x0: 8 bits scalar 0x1: 16 bits scalar 0x2: 32 bits scalar 0x3: Reserved	RW	0bxx

Table 16-50. Register Call Summary for Register DMA4_CSDPi

DMA_SYSTEM Functional Description

- Interrupt Generation: [0]
- Addressing Modes: [1]
- Packed Accesses: [2]
- Burst Transactions: [3]
- Endianism Conversion: [4] [5]
- Hardware Synchronization: [6]
- Graphics Acceleration Support: [7]
- Posted and Nonposted Writes: [8]
- Descriptors: [9]

DMA_SYSTEM Basic Programming Model

- Software-Triggered (Nonsynchronized) Transfer: [10] [11] [12] [13] [14] [15] [16] [17] [18]
- Hardware-Synchronized Transfer: [19] [20] [21] [22] [23] [24] [25] [26] [27] [28] [29] [30]
- 90-Degree Clockwise Image Rotation: [31] [32] [33] [34] [35] [36] [37] [38] [39]
- Linked-List Programming Guidelines: [40] [41]

DMA_SYSTEM Register Manual

- DMA_SYSTEM Register Summary: [42]

Table 16-51. DMA4_CENi

Address Offset	0x0000 0094 + (0x60 * i)	**index:**	i = 0 to 31
Physical Address	0x4A05 6094 + (0x60 * i)	**Instance**	DMA_SYSTEM
Description	Channel Element Number		
Type	RW		

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

Bits	Field Name	Description	Type	Reset
31:24	RESERVED	Write 0's for future compatibility. Read returns 0.	RW	0x00
23:0	CHANNEL_ELMNT_NBR	Number of elements within a frame (unsigned) to transfer	RW	0x------

Table 16-52. Register Call Summary for Register DMA4_CENi

DMA_SYSTEM Functional Description
- Interrupt Generation: [0]
- Addressing Modes: [1] [2]

DMA_SYSTEM Basic Programming Model
- Software-Triggered (Nonsynchronized) Transfer: [3]
- Hardware-Synchronized Transfer: [4] [5] [6]
- 90-Degree Clockwise Image Rotation: [7]

DMA_SYSTEM Register Manual
- DMA_SYSTEM Register Summary: [8]

Table 16-53. DMA4_CFNi

Address Offset	0x0000 0098 + (0x60 * i)	**index:**	i = 0 to 31
Physical Address	0x4A05 6098 + (0x60 * i)	**Instance**	DMA_SYSTEM
Description	Channel Frame Number		
Type	RW		

31–16	15–0
RESERVED	CHANNEL_FRAME_NBR

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Write 0's for future compatibility. Read returns 0.	RW	0x0000
15:0	CHANNEL_FRAME_NBR	Number of frames within the block to be transferred (unsigned)	RW	0x----

Table 16-54. Register Call Summary for Register DMA4_CFNi

DMA_SYSTEM Functional Description
- Interrupt Generation: [0]
- Addressing Modes: [1] [2]

DMA_SYSTEM Basic Programming Model
- Software-Triggered (Nonsynchronized) Transfer: [3]
- Hardware-Synchronized Transfer: [4] [5]
- 90-Degree Clockwise Image Rotation: [6]
- Linked-List Programming Guidelines: [7]

DMA_SYSTEM Register Manual
- DMA_SYSTEM Register Summary: [8]

Table 16-55. DMA4_CSSAi

Address Offset	0x0000 009C + (0x60 * i)	**index:**	i = 0 to 31
Physical Address	0x4A05 609C + (0x60 * i)	**Instance**	DMA_SYSTEM
Description	Channel Source Start Address		
Type	RW		

31–0
SRC_START_ADRS

Bits	Field Name	Description	Type	Reset
31:0	SRC_START_ADRS	32 bits of the source start address	RW	0x--------

Table 16-56. Register Call Summary for Register DMA4_CSSAi

DMA_SYSTEM Functional Description
- Addressing Modes: [0]

DMA_SYSTEM Basic Programming Model
- Software-Triggered (Nonsynchronized) Transfer: [1]
- Hardware-Synchronized Transfer: [2] [3] [4] [5]
- 90-Degree Clockwise Image Rotation: [6]

DMA_SYSTEM Register Manual
- DMA_SYSTEM Register Summary: [7]

Table 16-57. DMA4_CDSAi

Address Offset	0x0000 00A0 + (0x60 * i)	**index:**	i = 0 to 31
Physical Address	0x4A05 60A0 + (0x60 * i)	**Instance**	DMA_SYSTEM
Description	Channel Destination Start Address		
Type	RW		

31–0
DST_START_ADRS

Bits	Field Name	Description	Type	Reset
31:0	DST_START_ADRS	32 bits of the destination start address	RW	0x--------

Table 16-58. Register Call Summary for Register DMA4_CDSAi

DMA_SYSTEM Functional Description
- Addressing Modes: [0]

DMA_SYSTEM Basic Programming Model
- Software-Triggered (Nonsynchronized) Transfer: [1]
- Hardware-Synchronized Transfer: [2] [3] [4] [5]
- 90-Degree Clockwise Image Rotation: [6]

DMA_SYSTEM Register Manual
- DMA_SYSTEM Register Summary: [7]

Table 16-59. DMA4_CSEIi

Address Offset	0x0000 00A4 + (0x60 * i)	**index:**	i = 0 to 31
Physical Address	0x4A05 60A4 + (0x60 * i)	**Instance**	DMA_SYSTEM
Description	Channel Source Element Index (Signed)		
Type	RW		

31–16	15–0
RESERVED	CHANNEL_SRC_ELMNT_INDEX

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Write 0's for future compatibility. Read returns 0.	RW	0x0000
15:0	CHANNEL_SRC_ELMNT_INDEX	Channel source element index	RW	0x----

Table 16-60. Register Call Summary for Register DMA4_CSEIi

DMA_SYSTEM Functional Description
- Addressing Modes: [0]

DMA_SYSTEM Basic Programming Model
- Software-Triggered (Nonsynchronized) Transfer: [1]
- 90-Degree Clockwise Image Rotation: [2]

DMA_SYSTEM Register Manual
- DMA_SYSTEM Register Summary: [3]

Table 16-61. DMA4_CSFIi

Address Offset	0x0000 00A8 + (0x60 * i)	**index:**	i = 0 to 31
Physical Address	0x4A05 60A8 + (0x60 * i)	**Instance**	DMA_SYSTEM
Description	Channel Source Frame Index (Signed) or 16-bit Packet size		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH_SRC_FRM_INDEX_OR_16BIT_PKT_ELNT_NBR																															

Bits	Field Name	Description	Type	Reset
31:0	CH_SRC_FRM_INDEX_OR_16BIT_PKT_ELNT_NBR	Channel source frame index value if source address is in double index mode. Or if fs=bs=1 and DMA_CCR[SEL_SRC_DST_SYNC] = 1; the bit field [15:0] gives the number of element in packet. The field [31:16] is unused for the packet size.	RW	0x--------

Table 16-62. Register Call Summary for Register DMA4_CSFIi

DMA_SYSTEM Functional Description
- Addressing Modes: [0]
- Hardware Synchronization: [1]
- Packet Synchronization: [2] [3] [4] [5] [6]

DMA_SYSTEM Basic Programming Model
- Software-Triggered (Nonsynchronized) Transfer: [7]
- Hardware-Synchronized Transfer: [8] [9]
- 90-Degree Clockwise Image Rotation: [10]

DMA_SYSTEM Register Manual
- DMA_SYSTEM Register Summary: [11]

Table 16-63. DMA4_CDEIi

Address Offset	0x0000 00AC + (0x60 * i)	**index:**	i = 0 to 31
Physical Address	0x4A05 60AC + (0x60 * i)	**Instance**	DMA_SYSTEM
Description	Channel Destination Element Index (Signed)		
Type	RW		

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

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Write 0's for future compatibility. Read returns 0.	RW	0x0000
15:0	CHANNEL_DST_ELMNT_INDEX	Channel destination element index	RW	0x----

Table 16-64. Register Call Summary for Register DMA4_CDEIi

DMA_SYSTEM Functional Description
- Addressing Modes: [0]

DMA_SYSTEM Basic Programming Model
- Software-Triggered (Nonsynchronized) Transfer: [1]
- 90-Degree Clockwise Image Rotation: [2]

DMA_SYSTEM Register Manual
- DMA_SYSTEM Register Summary: [3]

Table 16-65. DMA4_CDFIi

Address Offset	0x0000 00B0 + (0x60 * i)	**index:**	i = 0 to 31
Physical Address	0x4A05 60B0 + (0x60 * i)	**Instance**	DMA_SYSTEM
Description	Channel Destination Frame Index (Signed) or 16-bit Packet size		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH_DST_FRM_IDX_OR_16BIT_PKT_ELNT_NBR																															

Bits	Field Name	Description	Type	Reset
31:0	CH_DST_FRM_IDX_OR_16BIT_PKT_ELNT_NBR	Channel destination frame index value if destination address is in double index mode. Or if fs=bs=1 and DMA_CCR[SEL_SRC_DST_SYNC]=0; the bit field [15:0] gives the number of element in packet. The field [31:16] is unused for the packet size..	RW	0x--------

Table 16-66. Register Call Summary for Register DMA4_CDFIi

DMA_SYSTEM Functional Description
- Addressing Modes: [0]
- Hardware Synchronization: [1]
- Packet Synchronization: [2] [3] [4] [5] [6]

DMA_SYSTEM Basic Programming Model
- Software-Triggered (Nonsynchronized) Transfer: [7]
- Hardware-Synchronized Transfer: [8] [9]
- 90-Degree Clockwise Image Rotation: [10]

DMA_SYSTEM Register Manual
- DMA_SYSTEM Register Summary: [11]

Table 16-67. DMA4_CSACi

Address Offset	0x0000 00B4 + (0x60 * i)	**index:**	i = 0 to 31
Physical Address	0x4A05 60B4 + (0x60 * i)	**Instance**	DMA_SYSTEM
Description	Channel Source Address Value. User has to access this register only in 32-bit access. If accessed in 8-bit or 16bit data may be corrupted.		
Type	R		

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

Bits	Field Name	Description	Type	Reset
31:0	SRC_ELMNT_ADRS	Current source address counter value	R	0x--------

Table 16-68. Register Call Summary for Register DMA4_CSACi

DMA_SYSTEM Register Manual
- DMA_SYSTEM Register Summary: [0]

Table 16-69. DMA4_CDACi

Address Offset	0x0000 00B8 + (0x60 * i)	**index:**	i = 0 to 31
Physical Address	0x4A05 60B8 + (0x60 * i)	**Instance**	DMA_SYSTEM
Description	Channel Destination Address Value. User has to access this register only in 32-bit access. If accessed in 8-bit or 16bit data may be corrupted.		
Type	RW		

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

Bits	Field Name	Description	Type	Reset
31:0	DST_ELMNT_ADRS	Current destination address counter value	RW	0x--------

Table 16-70. Register Call Summary for Register DMA4_CDACi

DMA_SYSTEM Functional Description
- Hardware Synchronization: [0]

DMA_SYSTEM Basic Programming Model
- Hardware-Synchronized Transfer: [1]
- Synchronized Transfer Monitoring Using CDAC: [2] [3] [4] [5] [6] [7] [8]

DMA_SYSTEM Register Manual
- DMA_SYSTEM Register Summary: [9]

Table 16-71. DMA4_CCENi

Address Offset	0x0000 00BC + (0x60 * i)	**index:**	i = 0 to 31
Physical Address	0x4A05 60BC + (0x60 * i)	**Instance**	DMA_SYSTEM
Description	Channel Current Transferred Element Number in the current frame. User has to access this register only in 32-bit access. If accessed in 8-bit or 16bit data may be corrupted.		
Type	RW		

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

Bits	Field Name	Description	Type	Reset
31:24	RESERVED	Write 0's for future compatibility. Read returns 0.	RW	0x00
23:0	CURRENT_ELMNT_NBR	Channel current transferred element number in the current frame	RW	0x------

Table 16-72. Register Call Summary for Register DMA4_CCENi

DMA_SYSTEM Functional Description
- Linked-List Control and Monitoring: [0] [1] [2]

DMA_SYSTEM Basic Programming Model
- Synchronized Transfer Monitoring Using CDAC: [3]

DMA_SYSTEM Register Manual
- DMA_SYSTEM Register Summary: [4]

Table 16-73. DMA4_CCFNi

Address Offset	0x0000 00C0 + (0x60 * i)	**index:**	i = 0 to 31
Physical Address	0x4A05 60C0 + (0x60 * i)	**Instance**	DMA_SYSTEM
Description	Channel Current Transferred Frame Number in the current transfer. User has to access this register only in 32-bit access. If accessed in 8-bit or 16bit data may be corrupted.		
Type	RW		

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

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Write 0's for future compatibility. Read returns 0.	RW	0x0000
15:0	CURRENT_FRAME_NBR	Channel current transferred frame number in the current transfer	RW	0x----

Table 16-74. Register Call Summary for Register DMA4_CCFNi

DMA_SYSTEM Functional Description
- Linked-List Control and Monitoring: [0] [1] [2]

DMA_SYSTEM Basic Programming Model
- Synchronized Transfer Monitoring Using CDAC: [3]

DMA_SYSTEM Register Manual
- DMA_SYSTEM Register Summary: [4]

Table 16-75. DMA4_COLORi

Address Offset	0x0000 00C4 + (0x60 * i)	**index:**	i = 0 to 31
Physical Address	0x4A05 60C4 + (0x60 * i)	**Instance**	DMA_SYSTEM
Description	Channel DMA COLOR KEY /SOLID COLOR		
Type	RW		

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

Bits	Field Name	Description	Type	Reset
31:24	RESERVED	Write 0's for future compatibility. Read returns 0.	RW	0x–
23:0	CH_BLT_FRGRND_COLOR_OR_SOLID_COLOR_PTRN	Color key or solid color pattern: The pattern is replicated according to the data type. If the data-type is 8-bit the pattern is replicated 4 times to fill the register in order to enhance processing when data is packed at the graphic module input. The same reasoning for 16-bit data-type.	RW	0x------

Table 16-76. Register Call Summary for Register DMA4_COLORi

DMA_SYSTEM Functional Description
- Graphics Acceleration Support: [0] [1]
- Descriptors: [2]

DMA_SYSTEM Basic Programming Model
- Graphic Operations: [3] [4]

DMA_SYSTEM Register Manual
- DMA_SYSTEM Register Summary: [5]

Table 16-77. DMA4_CDPi

Address Offset	0x0000 00D0 + (0x60 * i)	**index:**	i = 0 to 31
Physical Address	0x4A05 60D0 + (0x60 * i)	**Instance**	DMA_SYSTEM
Description	This register controls the various parameters of the link list mechanism		
Type	RW		

31:11	10	9:8	7	6:4	3:2	1:0
RESERVED	FAST	TRANSFER_MODE	PAUSE_LINK_LIST	NEXT_DESCRIPTOR_TYPE	SRC_VALID	DEST_VALID

Bits	Field Name	Description	Type	Reset
31:11	RESERVED	Write 0's for future compatibility, Reads return 0	RW	0x00000
10	FAST	Sets the fast-start mode for linked list descriptor types 1, 2 and 3 0x0: No fast-start mode 0x1: Fast-start mode is enabled.	RW	0x0
9:8	TRANSFER_MODE	Enable linked-list transfer mode 0x0: Normal transfer mode is used. 0x1: Linked-list channel mode for type 1, 2, or 3 descriptor is used. 0x2: Undefined 0x3: Undefined	RW	0x0
7	PAUSE_LINK_LIST	Suspend the linked-list transfer at completion of the current block transfer. 0x0: Linked list is active. 0x1: Linked list is suspended at the boundary of next descriptor loading.	RW	0x0
6:4	NEXT_DESCRIPTOR_TYPE	Next Descriptor Type 0x0: Undefined 0x1: Next descriptor is of type 1. 0x2: Next descriptor is of type 2. 0x3: Next descriptor is of type 3. 0x4: Undefined 0x5: Undefined 0x6: Undefined 0x7: Undefined	RW	0x-
3:2	SRC_VALID	Source address valid 0x0: The source address is not present in the next descriptor and continuous incrementing is enabled. 0x1: The source address must be reloaded in the next descriptor transfer. 0x2: The source start address is not present in the next descriptor. But will reload the one from configuration memory which belongs to the previous descriptor. 0x3: Undefined addressing mode	RW	0x-

Bits	Field Name	Description	Type	Reset
1:0	DEST_VALID	Destination address valid 0x0: The destination address is not present in the next descriptor and continuous incrementing is enabled. 0x1: The destination address must be reloaded in the next descriptor transfer. 0x2: The destination start address is not present in the next descriptor. But will reload the one from configuration memory which belongs to the previous descriptor. 0x3: Undefined addressing mode	RW	0x-

Table 16-78. Register Call Summary for Register DMA4_CDPi

DMA_SYSTEM Functional Description
- Link-List Transfer Profile: [0]
- Descriptors: [1] [2]
- Linked-List Control and Monitoring: [3] [4] [5] [6] [7] [8] [9] [10] [11]

DMA_SYSTEM Basic Programming Model
- Linked-List Programming Guidelines: [12]

DMA_SYSTEM Register Manual
- DMA_SYSTEM Register Summary: [13]

Table 16-79. DMA4_CNDPi

Address Offset	0x0000 00D4 + (0x60 * i)	**index:**	i = 0 to 31
Physical Address	0x4A05 60D4 + (0x60 * i)	**Instance**	DMA_SYSTEM
Description	This register contains the Next descriptor Address Pointer for the link list Mechanism		
Type	RW		

31 … 2	1 0
NEXT_DESCRIPTOR_POINTER	RESERVED

Bits	Field Name	Description	Type	Reset
31:2	NEXT_DESCRIPTOR_POINTER	This register contains the Next descriptor Address Pointer for the link list Mechanism	RW	0bxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
1:0	RESERVED	Write 0's for future compatibility, Reads return 0	RW	0x0

Table 16-80. Register Call Summary for Register DMA4_CNDPi

DMA_SYSTEM Functional Description
- Link-List Transfer Profile: [0]

DMA_SYSTEM Register Manual
- DMA_SYSTEM Register Summary: [1]

Table 16-81. DMA4_CCDNi

Address Offset	0x0000 00D8 + (0x60 * i)	**index:**	i = 0 to 31
Physical Address	0x4A05 60D8 + (0x60 * i)	**Instance**	DMA_SYSTEM
Description			
Type	RW		

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

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Write 0's for future compatibility, Reads return 0	RW	0x0000
15:0	CURRENT_DESCRIPTOR_NBR	This register when read contains the current active descriptor number in the link list. This register is Read/write to allow user initialization.	RW	0x----

Table 16-82. Register Call Summary for Register DMA4_CCDNi

DMA_SYSTEM Functional Description

- Linked-List Control and Monitoring: [0] [1] [2]

DMA_SYSTEM Register Manual

- DMA_SYSTEM Register Summary: [3]

16.2 Enhanced DMA

This chapter describes the Enhanced Direct Memory Access (EDMA) controller.

NOTE: EVE is not supported in this family of devices.

16.2.1 EDMA Module Overview

The enhanced direct memory access module, also called EDMA, performs high-performance data transfers between two slave points, memories and peripheral devices without microprocessor unit (MPU) or digital signal processor (DSP) support during transfer. EDMA transfer is programmed through a logical EDMA channel, which allows the transfer to be optimally tailored to the requirements of the application.

The EDMA can also perform transfers between external memories and between Device subsystems internal memories, with some performance loss caused by resource sharing between the read and write ports.

EDMA controller is based on two major principal blocks:

- EDMA third-party channel controller (EDMA_TPCC)
- EDMA third-party transfer controller (EDMA_TPTC)

The **TPCC** is a high flexible Channel Controller. It serves as an user interface and an event interface for the EDMA controller. The EDMA_TPCC serves to prioritize incoming software requests or events from peripherals and submits transfer requests (TRs) to the transfer controller.

The **TPTC** performs read and write transfers by EDMA ports to the slave peripherals as programmed in the "Active" and "Pending" set of the registers. The transfer controllers are responsible for data movement and issue read/write commands to the source and destination addresses that are programmed for a given transfer in the EDMA_TPCC.

The SoC integrates the following EDMA instances:

- One system-level EDMA
- One DSP internal EDMA (per DSP)
- One EVE internal EDMA (per EVE)

Each of these EDMA modules consists of:

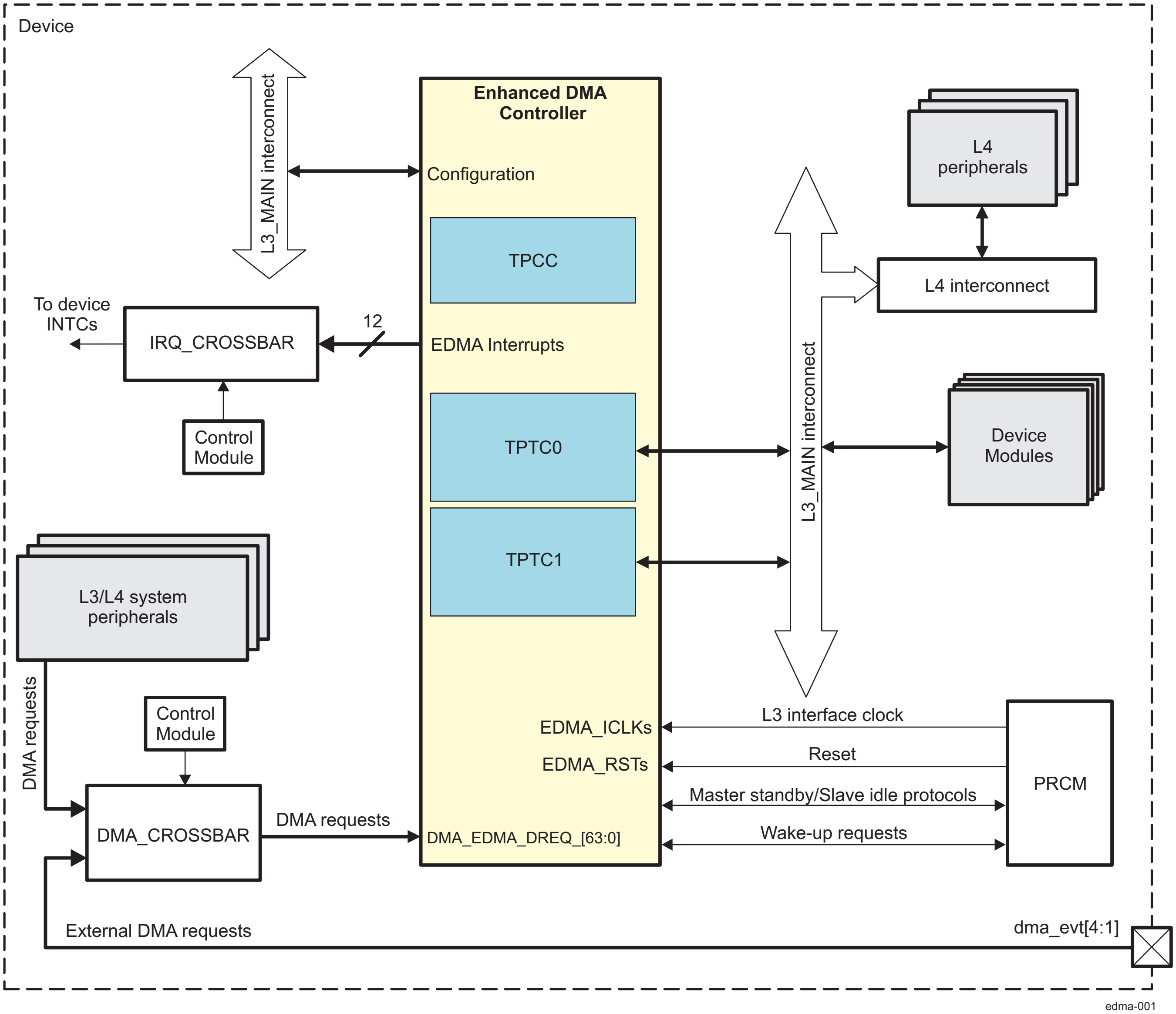
- One TPCC instance
- Two TPTC instances

NOTE: All EDMA modules in the SoC are functionally identical. Note that some of the configuration parameters may be different for the various EDMA instances (see Section 16.2.2, *EDMA Controllers Configuration*).

This chapter is mostly focused on describing the system-level EDMA module (in terms of configuration and integration in the SoC). For details on DSPx_EDMA / EVEx_EDMA integration, see their respective chapters.

Figure 16-12 shows an overview of the EDMA module.

Figure 16-12. EDMA module Overview



The device CPUs can configure the EDMA controller blocks through the L3_MAIN interconnect.

16.2.1.1 EDMA Features

The **EDMA_TPCC** channel controller has following features:

- Fully orthogonal transfer description:
  - Three transfer dimensions.
  - A-synchronized transfers: one-dimension serviced per event.
  - AB-synchronized transfers: two-dimensions serviced per event.
  - Independent indexes on source and destination.
  - Chaining feature allows a 3-D transfer based on a single event.
- Flexible transfer definition:
  - Increment or FIFO transfer addressing modes.
  - Linking mechanism allows automatic PaRAM set update.

  - Chaining allows multiple transfers to execute with one event.
- Interrupt generation for the following:
  - Transfer completion.
  - Error conditions.
- Debug visibility:
  - Queue water marking/threshold.
  - Error and status recording to facilitate debug.
- 64 DMA request channels:
  - Event synchronization.
  - Manual synchronization (CPU(s) write to event set registers EDMA_TPCC_ESR and EDMA_TPCC_ESRH).
  - Chain synchronization (completion of one transfer triggers another transfer).
- Eight QDMA channels:
  - QDMA channels trigger automatically upon writing to a parameter RAM (PaRAM) set entry.
  - Support for programmable QDMA channel to PaRAM mapping.
- 512 PaRAM sets:
  - Each PaRAM set can be used for a DMA channel, QDMA channel, or link set.
- Two transfer controllers/event queues.
- 16 event entries per event queue.
- Memory protection support:
  - Proxy memory protection for TR submission.
  - Active memory protection for accesses to PaRAM and registers.

The **EDMA_TPTC** transfer controller has the following features:

- Two transfer controllers (TC).
- 128-bit wide read and write ports per TC.
- Up to four in-flight transfer requests (TRs).
- Programmable priority level.
- Supports two-dimensional transfers with independent indexes on source and destination (EDMA_TPCC manages the 3rd dimension).
- Support for increment or constant addressing mode transfers.
- Interrupt and error support.
- Memory-Mapped Register (MMR) bit fields are fixed position in 32-bit MMR regardless of endianness.

EDMA controller uses the shared MMU1 module for transfering to and from DSP module. This provides several benefits including:

- Protection of Host CPU memory regions from accidental corruption by EDMA TPTCs.
- Direct allocation of buffers in user space without the need for translation between CPU and DSP applications utilizing EDMA TPTCs.

Accesses by the EDMA TPTCs (both TPTC0 and TPTC1) may optionally be routed through the MMU1.

The TPTC0 and TPTC1 routing allows EDMA transfer controller to be used to perform transfers using only the virtual addresses of the associated buffers.

For more information about MMU1 module refer to Chapter 20 *Memory Management Units*.

16.2.2 EDMA Controllers Configuration

Table 16-83 summarizes the configuration for each of the EDMA channel controllers present on the SoC.

Table 16-83. EDMA Channel Controllers Configuration

Parameter	SYS_EDMA CC Configuration	DSPx_EDMA CC Configuration	EVEx_EDMA CC Configuration
Number of DMA channels (NUM_DMACH)	64	64	16
Number of QDMA channels (NUM_QDMACH)	8	8	8
Number of interrupt channels (NUM_INTCH)	64	64	16
Number of PaRAM set entries (NUM_PARAMENTRY)	512	128	128
Number of event queues (NUM_EVQUE)	2	2	2
Number of transfer controllers (NUM_TC)	2	2	2
Memory protection existence (MPEXIST)	Yes	Yes	Yes
Number of memory protection and shadow regions (NUM_REGIONS)	8	8	8
Channel mapping existence (CHMAPEXIST)	Yes	Yes	Yes

Table 16-84 summarizes the configuration of each of the EDMA transfer controllers present on the SoC.

Table 16-84. EDMA Transfer Controllers Configuration

Parameter	SYS_EDMA TC0 / TC1 Configuration	DSPx_EDMA TC0 / TC1 Configuration	EVEx_EDMA TC0 / TC1 Configuration
Data FIFO size (FIFOSIZE)	1024 bytes	2048 bytes	2048 bytes
Bus width (BUSBYTE)	16 bytes	16 bytes	16 bytes
Number of destination FIFO register sets (DSTREGDEPTH)	4 entries	4 entries	4 entries
Default burst size (DBS)	Defined by CTRL_CORE_CONTROL_IO_1 register	Defined by DSP_SYS_BUS_CONFIG register	Defined by EVE_BUS_CONFIG register

16.2.3 EDMA Controller Environment

The EDMA controller supports external DMA requests through the dma_evt[4:1] pins (see Table 16-85). A logical channel can be configured to respond to an external synchronization request.

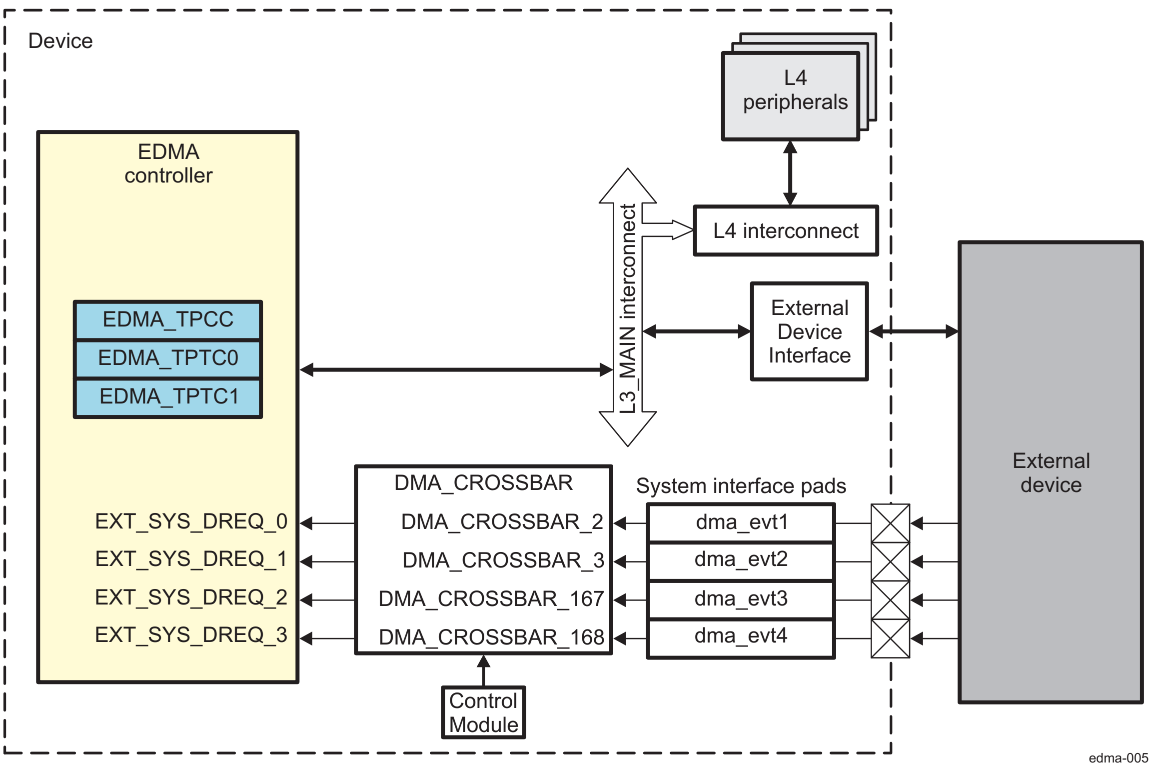
Table 16-85. External EDMA Request Signals

Pin Name	DMA_CROSSBAR Input	Signal Name	I/O [(1)]	Description	Module Reset Value
dma_evt1	DMA_CROSSBAR_2	EXT_SYS_DREQ_0	I	External DMA request 0 (system expansion)	Z
dma_evt2	DMA_CROSSBAR_3	EXT_SYS_DREQ_1	I	External DMA request 1 (system expansion)	Z
dma_evt3	DMA_CROSSBAR_167	EXT_SYS_DREQ_2	I	External DMA request 2 (system expansion)	Z
dma_evt4	DMA_CROSSBAR_168	EXT_SYS_DREQ_3	I	External DMA request 3 (system expansion)	Z

(1) I = Input, O = Output

Figure 16-13 shows an example of how to use the external hardware DMA request pins in the EDMA environment.

Figure 16-13. Example of External DMA Requests Use



An external device can use the external DMA request pins to start a logical channel transfer. The transfer can be a memory-to-memory transfer in which the source memory is in the external device.

By default, the external DMA request signals are not available on external pins after a cold reset. For more information about multiplexing out the two signal lines to pins, refer to Chapter 18, *Control Module*.

All 64 DMA request lines are transition sensitive.

16.2.4 EDMA Controller Integration

This section describes the integration of the module in the device, including information about clocks, resets, and hardware requests.

Figure 16-14 shows the EDMA controller integration.

Figure 16-14. EDMA Controller Integration

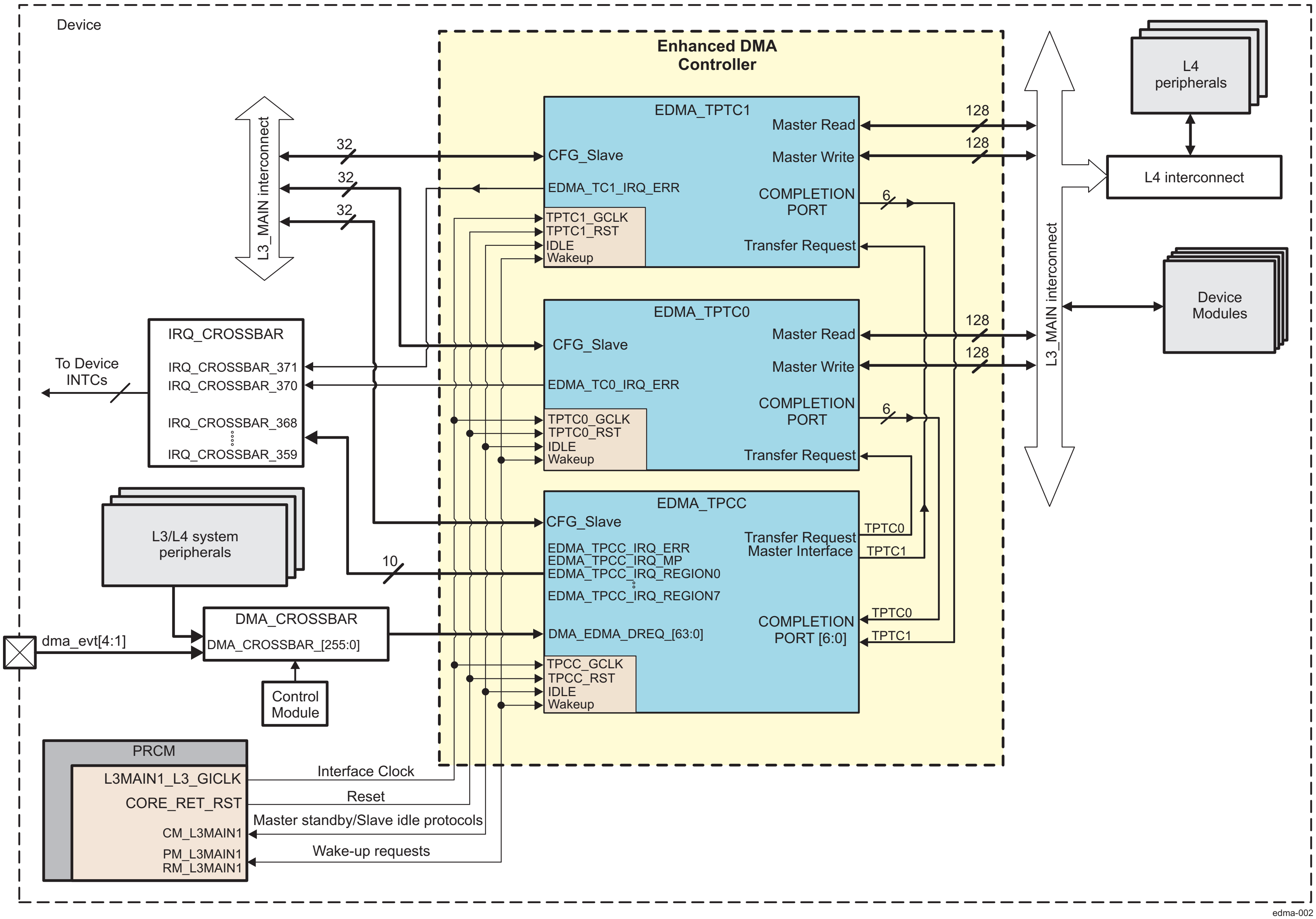


Table 16-86 through Table 16-88 summarize the integration of the module in the device.

Table 16-86. EDMA Integration Attributes

Module Instance	Attributes	
	Power Domain	Interconnect
EDMA_TPCC	PD_COREAON	L3_MAIN
EDMA_TPTC0		
EDMA_TPTC1		

Table 16-87. EDMA Clocks and Resets

Clocks				
Module Instance	**Destination Signal Name**	**Source Signal Name**	**Source**	**Description**
EDMA_TPCC	EDMA_TPCC_GCLK	L3MAIN1_L3_GICLK	PRCM	Interface clock. It supports the configuration port. For information about PRCM clock gating and management, see Chapter 3, *Power, Reset, and Clock Management*.
EDMA_TPTC0	EDMA_TPTC0_GCLK			
EDMA_TPTC1	EDMA_TPTC1_GCLK			
Resets				
Module Instance	**Destination Signal Name**	**Source Signal Name**	**Source**	**Description**
EDMA_TPCC	EDMA_TPCC_RST	CORE_RET_RST	PRCM	Hardware retention reset. It initializes all internal logic of the EDMA contrller modules, all global registers, and some of the per-channel registers, implemented in flip-flops. However, all remaining per-channel registers are memory-based, and, therefore, are not reset (have undefined values). Thus, when programming a channel for the first time, all bits that have undefined reset values must be configured before enabling the channel. For information about PRCM reset sources and distribution, see Chapter 3, *Power, Reset, and Clock Management*.
EDMA_TPTC0	EDMA_TPTC0_RST			
EDMA_TPTC1	EDMA_TPTC1_RST			

Table 16-88. EDMA Hardware Requests

Interrupt Requests				
Module Instance	**Source Signal Name**	**Destination IRQ_CROSSBAR INPUT**	**Default mapping**	**Description**
EDMA_TPCC	EDMA_TPCC_IRQ_ERR	IRQ_CROSSBAR_359	-	TPCC error interrupt. This IRQ source signal is not mapped by default to any device INTC. For more information about INTC refer to Chapter 17 *Interrupt Controllers*.
	EDMA_TPCC_IRQ_MP	IRQ_CROSSBAR_360	-	TPCC memory protection interrupt. This IRQ source signal is not mapped by default to any device INTC.

Table 16-88. EDMA Hardware Requests (continued)

	EDMA_TPCC_IRQ_REGION0	IRQ_CROSSBAR_361	-	TPCC Region 0 interrupt. This IRQ source signal is not mapped by default to any device INTC.
	EDMA_TPCC_IRQ_REGION1	IRQ_CROSSBAR_362	-	TPCC Region 1 interrupt. This IRQ source signal is not mapped by default to any device INTC.
	EDMA_TPCC_IRQ_REGION2	IRQ_CROSSBAR_363	-	TPCC Region 2 interrupt. This IRQ source signal is not mapped by default to any device INTC.
	EDMA_TPCC_IRQ_REGION3	IRQ_CROSSBAR_364	-	TPCC Region 3 interrupt. This IRQ source signal is not mapped by default to any device INTC.
	EDMA_TPCC_IRQ_REGION4	IRQ_CROSSBAR_365	-	TPCC Region 4 interrupt. This IRQ source signal is not mapped by default to any device INTC.
	EDMA_TPCC_IRQ_REGION5	IRQ_CROSSBAR_366	-	TPCC Region 5 interrupt. This IRQ source signal is not mapped by default to any device INTC.
	EDMA_TPCC_IRQ_REGION6	IRQ_CROSSBAR_367	-	TPCC Region 6 interrupt. This IRQ source signal is not mapped by default to any device INTC.
	EDMA_TPCC_IRQ_REGION7	IRQ_CROSSBAR_368	-	TPCC Region 7 interrupt. This IRQ source signal is not mapped by default to any device INTC.
EDMA_TPTC0	EDMA_TC0_IRQ_ERR	IRQ_CROSSBAR_370	-	TPTC0 error interrupt. This IRQ source signal is not mapped by default to any device INTC.
EDMA_TPTC1	EDMA_TC1_IRQ_ERR	IRQ_CROSSBAR_371	-	TPTC1 error interrupt. This IRQ source signal is not mapped by default to any device INTC.

NOTE: The “**Default Mapping**” column in Table 16-88 *EDMA Hardware Requests* shows the default mapping of module IRQ source signals. These IRQ source signals can also be mapped to other lines of each device Interrupt controller through the IRQ_CROSSBAR or DMA_CROSSBAR modules.
For more information about the IRQ_CROSSBAR and DMA_CROSSBAR modules, see sections: Section 18.4.6.4 *IRQ_CROSSBAR Module Functional Description* and Section 18.4.6.5 *DMA_CROSSBAR Module Functional Description*, in Chapter 18 *Control Module*.
For more information about the device interrupt controllers, see Chapter 17 *Interrupt Controllers*.

NOTE: For a description of the interrupt source, see Section 16.2.5.9, *EDMA interrupts*.

16.2.4.1 EDMA Requests to the EDMA Controller

Table 16-89 lists the default DMA sources for the EDMA controller. In addition, the EDMA inputs (DMA_EDMA_DREQ_[63:0]) can alternatively be sourced through the associated DMA_CROSSBAR from one of the 256 multiplexed device DMA sources listed in Table 16-6. The CTRL_CORE_DMA_EDMA_DREQ_y_z registers (where *y* and *z* are indexes of EDMA input lines) in the Control Module are used to select between the default DMA sources and the multiplexed DMA sources.

Table 16-89. EDMA Default Request Mapping

DMA Request Line	DMA_ CROSSBAR Instance Number	DMA_CROSSBAR Configuration Register	DMA_ CROSSBAR Default Input Index	Default DMA Source Name	Default DMA Source Description
DMA_EDMA_DREQ_0	1	CTRL_CORE_DMA_EDMA_DREQ_0_1[7:0]	1	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_EDMA_DREQ_1	2	CTRL_CORE_DMA_EDMA_DREQ_0_1[23:16]	2	EXT_SYS_DREQ_0	External DMA request 0 (system expansion)
DMA_EDMA_DREQ_2	3	CTRL_CORE_DMA_EDMA_DREQ_2_3[7:0]	3	EXT_SYS_DREQ_1	External DMA request 1 (system expansion)
DMA_EDMA_DREQ_3	4	CTRL_CORE_DMA_EDMA_DREQ_2_3[23:16]	4	GPMC_DREQ	GPMC data transmit request from prefetch engine
DMA_EDMA_DREQ_4	5	CTRL_CORE_DMA_EDMA_DREQ_4_5[7:0]	5	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_EDMA_DREQ_5	6	CTRL_CORE_DMA_EDMA_DREQ_4_5[23:16]	6	DISPC_DREQ	Frame update request
DMA_EDMA_DREQ_6	7	CTRL_CORE_DMA_EDMA_DREQ_6_7[7:0]	7	CT_TBR_DREQ	DEBUG subsystem CT_TBR request
DMA_EDMA_DREQ_7	8	CTRL_CORE_DMA_EDMA_DREQ_6_7[23:16]	8	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_EDMA_DREQ_8	9	CTRL_CORE_DMA_EDMA_DREQ_8_9[7:0]	9	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_EDMA_DREQ_9	10	CTRL_CORE_DMA_EDMA_DREQ_8_9[23:16]	10	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_EDMA_DREQ_10	11	CTRL_CORE_DMA_EDMA_DREQ_10_11[7:0]	11	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_EDMA_DREQ_11	12	CTRL_CORE_DMA_EDMA_DREQ_10_11[23:16]	12	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_EDMA_DREQ_12	13	CTRL_CORE_DMA_EDMA_DREQ_12_13[7:0]	13	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_EDMA_DREQ_13	14	CTRL_CORE_DMA_EDMA_DREQ_12_13[23:16]	14	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_EDMA_DREQ_14	15	CTRL_CORE_DMA_EDMA_DREQ_14_15[7:0]	15	MCSPI3_DREQ_TX0	McSPI3 transmit request channel 0
DMA_EDMA_DREQ_15	16	CTRL_CORE_DMA_EDMA_DREQ_14_15[23:16]	16	MCSPI3_DREQ_RX0	McSPI3 receive request channel 0
DMA_EDMA_DREQ_16	17	CTRL_CORE_DMA_EDMA_DREQ_16_17[7:0]	17	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_EDMA_DREQ_17	18	CTRL_CORE_DMA_EDMA_DREQ_16_17[23:16]	18	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_EDMA_DREQ_18	19	CTRL_CORE_DMA_EDMA_DREQ_18_19[7:0]	19	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_EDMA_DREQ_19	20	CTRL_CORE_DMA_EDMA_DREQ_18_19[23:16]	20	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_EDMA_DREQ_20	21	CTRL_CORE_DMA_EDMA_DREQ_20_21[7:0]	21	Reserved	Reserved by default but can be remapped to a valid DMA source

Table 16-89. EDMA Default Request Mapping (continued)

DMA Request Line	DMA_ CROSSBAR Instance Number	DMA_CROSSBAR Configuration Register	DMA_ CROSSBAR Default Input Index	Default DMA Source Name	Default DMA Source Description
DMA_EDMA_DREQ_21	22	CTRL_CORE_DMA_EDMA_DREQ_20_21[23:16]	22	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_EDMA_DREQ_22	23	CTRL_CORE_DMA_EDMA_DREQ_22_23[7:0]	23	MCSPI3_DREQ_TX1	McSPI3 transmit request channel 1
DMA_EDMA_DREQ_23	24	CTRL_CORE_DMA_EDMA_DREQ_22_23[23:16]	24	MCSPI3_DREQ_RX1	McSPI3 receive request channel 1
DMA_EDMA_DREQ_24	25	CTRL_CORE_DMA_EDMA_DREQ_24_25[7:0]	25	I2C3_DREQ_TX	I2C3 transmit request
DMA_EDMA_DREQ_25	26	CTRL_CORE_DMA_EDMA_DREQ_24_25[23:16]	26	I2C3_DREQ_RX	I2C3 receive request
DMA_EDMA_DREQ_26	27	CTRL_CORE_DMA_EDMA_DREQ_26_27[7:0]	27	I2C1_DREQ_TX	I2C1 transmit request
DMA_EDMA_DREQ_27	28	CTRL_CORE_DMA_EDMA_DREQ_26_27[23:16]	28	I2C1_DREQ_RX	I2C1 receive request
DMA_EDMA_DREQ_28	29	CTRL_CORE_DMA_EDMA_DREQ_28_29[7:0]	29	I2C2_DREQ_TX	I2C2 transmit request
DMA_EDMA_DREQ_29	30	CTRL_CORE_DMA_EDMA_DREQ_28_29[23:16]	30	I2C2_DREQ_RX	I2C2 receive request
DMA_EDMA_DREQ_30	31	CTRL_CORE_DMA_EDMA_DREQ_30_31[7:0]	31	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_EDMA_DREQ_31	32	CTRL_CORE_DMA_EDMA_DREQ_30_31[23:16]	32	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_EDMA_DREQ_32	33	CTRL_CORE_DMA_EDMA_DREQ_32_33[7:0]	33	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_EDMA_DREQ_33	34	CTRL_CORE_DMA_EDMA_DREQ_32_33[23:16]	34	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_EDMA_DREQ_34	35	CTRL_CORE_DMA_EDMA_DREQ_34_35[7:0]	35	MCSPI1_DREQ_TX0	McSPI1 transmit request channel 0
DMA_EDMA_DREQ_35	36	CTRL_CORE_DMA_EDMA_DREQ_34_35[23:16]	36	MCSPI1_DREQ_RX0	McSPI1 receive request channel 0
DMA_EDMA_DREQ_36	37	CTRL_CORE_DMA_EDMA_DREQ_36_37[7:0]	37	MCSPI1_DREQ_TX1	McSPI1 transmit request channel 1
DMA_EDMA_DREQ_37	38	CTRL_CORE_DMA_EDMA_DREQ_36_37[23:16]	38	MCSPI1_DREQ_RX1	McSPI1 receive request channel 1
DMA_EDMA_DREQ_38	39	CTRL_CORE_DMA_EDMA_DREQ_38_39[7:0]	39	MCSPI1_DREQ_TX2	McSPI1 transmit request channel 2
DMA_EDMA_DREQ_39	40	CTRL_CORE_DMA_EDMA_DREQ_38_39[23:16]	40	MCSPI1_DREQ_RX2	McSPI1 receive request channel 2
DMA_EDMA_DREQ_40	41	CTRL_CORE_DMA_EDMA_DREQ_40_41[7:0]	41	MCSPI1_DREQ_TX3	McSPI1 transmit request channel 3
DMA_EDMA_DREQ_41	42	CTRL_CORE_DMA_EDMA_DREQ_40_41[23:16]	42	MCSPI1_DREQ_RX3	McSPI1 receive request channel 3
DMA_EDMA_DREQ_42	43	CTRL_CORE_DMA_EDMA_DREQ_42_43[7:0]	43	MCSPI2_DREQ_TX0	McSPI2 transmit request channel 0
DMA_EDMA_DREQ_43	44	CTRL_CORE_DMA_EDMA_DREQ_42_43[23:16]	44	MCSPI2_DREQ_RX0	McSPI2 receive request channel 0
DMA_EDMA_DREQ_44	45	CTRL_CORE_DMA_EDMA_DREQ_44_45[7:0]	45	MCSPI2_DREQ_TX1	McSPI2 transmit request channel 1
DMA_EDMA_DREQ_45	46	CTRL_CORE_DMA_EDMA_DREQ_44_45[23:16]	46	MCSPI2_DREQ_RX1	McSPI2 receive request channel 1
DMA_EDMA_DREQ_46	47	CTRL_CORE_DMA_EDMA_DREQ_46_47[7:0]	47	MMC2_DREQ_TX	MMC2 transmit request
DMA_EDMA_DREQ_47	48	CTRL_CORE_DMA_EDMA_DREQ_46_47[23:16]	48	MMC2_DREQ_RX	MMC2 receive request
DMA_EDMA_DREQ_48	49	CTRL_CORE_DMA_EDMA_DREQ_48_49[7:0]	49	UART1_DREQ_TX	UART1 transmit request

Table 16-89. EDMA Default Request Mapping (continued)

DMA Request Line	DMA_ CROSSBAR Instance Number	DMA_CROSSBAR Configuration Register	DMA_ CROSSBAR Default Input Index	Default DMA Source Name	Default DMA Source Description
DMA_EDMA_DREQ_49	50	CTRL_CORE_DMA_EDMA_DREQ_48_49[23:16]	50	UART1_DREQ_RX	UART1 receive request
DMA_EDMA_DREQ_50	51	CTRL_CORE_DMA_EDMA_DREQ_50_51[7:0]	51	UART2_DREQ_TX	UART2 transmit request
DMA_EDMA_DREQ_51	52	CTRL_CORE_DMA_EDMA_DREQ_50_51[23:16]	52	UART2_DREQ_RX	UART2 receive request
DMA_EDMA_DREQ_52	53	CTRL_CORE_DMA_EDMA_DREQ_52_53[7:0]	53	UART3_DREQ_TX	UART3 transmit request
DMA_EDMA_DREQ_53	54	CTRL_CORE_DMA_EDMA_DREQ_52_53[23:16]	54	UART3_DREQ_RX	UART3 receive request
DMA_EDMA_DREQ_54	55	CTRL_CORE_DMA_EDMA_DREQ_54_55[7:0]	55	UART4_DREQ_TX	UART4 transmit request
DMA_EDMA_DREQ_55	56	CTRL_CORE_DMA_EDMA_DREQ_54_55[23:16]	56	UART4_DREQ_RX	UART4 receive request
DMA_EDMA_DREQ_56	57	CTRL_CORE_DMA_EDMA_DREQ_56_57[7:0]	57	MMC4_DREQ_TX	MMC4 transmit request
DMA_EDMA_DREQ_57	58	CTRL_CORE_DMA_EDMA_DREQ_56_57[23:16]	58	MMC4_DREQ_RX	MMC4 receive request
DMA_EDMA_DREQ_58	59	CTRL_CORE_DMA_EDMA_DREQ_58_59[7:0]	59	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_EDMA_DREQ_59	60	CTRL_CORE_DMA_EDMA_DREQ_58_59[23:16]	60	Reserved	Reserved by default but can be remapped to a valid DMA source
DMA_EDMA_DREQ_60	61	CTRL_CORE_DMA_EDMA_DREQ_60_61[7:0]	61	MMC1_DREQ_TX	MMC1 transmit request
DMA_EDMA_DREQ_61	62	CTRL_CORE_DMA_EDMA_DREQ_60_61[23:16]	62	MMC1_DREQ_RX	MMC1 receive request
DMA_EDMA_DREQ_62	63	CTRL_CORE_DMA_EDMA_DREQ_62_63[7:0]	63	UART5_DREQ_TX	UART5 transmit request
DMA_EDMA_DREQ_63	64	CTRL_CORE_DMA_EDMA_DREQ_62_63[23:16]	64	UART5_DREQ_RX	UART5 receive request

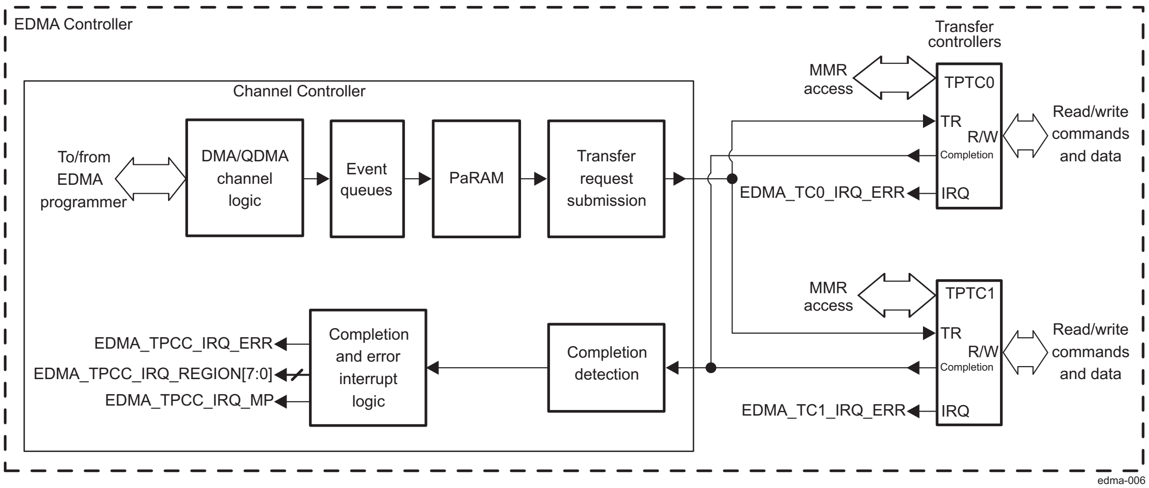
16.2.5 EDMA Controller Functional Description

This chapter discusses the architecture of the EDMA controller.

16.2.5.1 Block Diagram

Figure 16-15 shows the functional block diagram of the EDMA controller.

Figure 16-15. EDMA Controller Block Diagram



16.2.5.1.1 Third-Party Channel Controller

The TPCC is the EDMA transfer scheduler responsible for scheduling, arbitrating, and issuing user programmed transfers to the two TPTCs.

Figure 16-16 shows a functional block diagram of the EDMA channel controller (EDMA_TPCC).

The main blocks of the EDMA_TPCC are as follows:

- Parameter RAM (PaRAM): The PaRAM maintains parameter sets for channel and reload parameter sets. The PaRAM must be written with the transfer context for the desired channels and link parameter sets. EDMA_TPCC processes and sets based on a trigger event and submits a transfer request (TR) to the transfer controllers.
- EDMA event and interrupt processing registers: Allows mapping of events to parameter sets, enable/disable events, enable/disable interrupt conditions, and clearing interrupts.
- Completion detection: The completion detect block detects completion of transfers by the EDMA_TPTCs or slave peripherals. The completion of transfers can be used optionally to chain trigger new transfers or to assert interrupts.
- Event queues: Event queues form the interface between the event detection logic and the transfer request submission logic.
- Memory protection registers: Memory protection registers define the accesses (privilege level and requestor(s)) that are allowed to access the DMA channel shadow region view(s) and regions of PaRAM.

Other functions include the following:

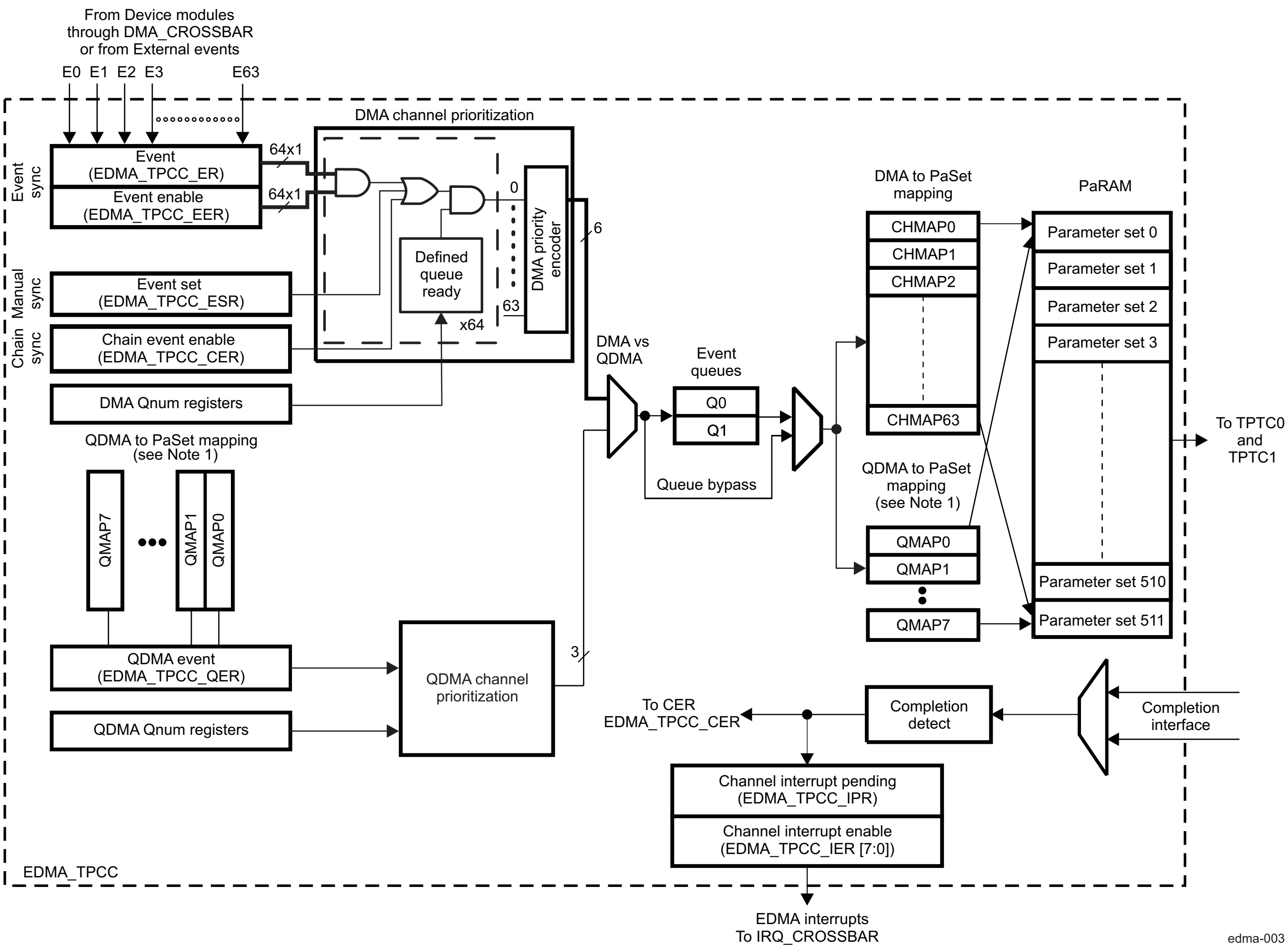
- Region registers: Region registers allow DMA resources (DMA channels and interrupts) to be assigned to unique regions that different EDMA programmers own (for example, MPU or DSPs).
- Debug registers: Debug registers allow debug visibility by providing registers to read the queue status, controller status, and missed event status.

The EDMA_TPCC includes two channel types: DMA channels (64 channels) and QDMA channels (8 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 a QDMA channel is the method that the system uses to trigger transfers.

Figure 16-16 is a block diagram of the EDMA_TPCC.

Figure 16-16. EDMA Channel Controller Block Diagram



(1) Although it is depicted twice in Figure 16-16, there is only one physical register set for the QDMA to PaRAM set mapping block.

The EDMA_TPCC supports up to 64 DMA channels and up to 8 QDMA channels. These channels are identical, except for how they are triggered:

- DMA channels are triggered by external events (such as McSPI modules TX event and McSPI modules RX event) by the event set registers EDMA_TPCC_ESR and EDMA_TPCC_ESRH, or through chaining register EDMA_TPCC_CER.
- QDMA channels are triggered automatically (auto-triggered) by the CPU. QDMAs allow a minimum number of linear writes to be issued to the TPCC to force a series of transfers to occur.

The TPCC arbitrates among pending DMA and QDMA events with a fixed [64:1] and [8:1] priority encoder for these events, respectively (a low channel number corresponds to a high priority).

DMA events are always higher priority than QDMA events. The higher-priority event is placed in the event queue to await submission to the transfer controllers, which occurs at the earliest opportunity. Each event queue is serviced in FIFO order, with a maximum of 16 queued events per event queue. If more than one TPTC is ready to be programmed with a transmission request (TR), the event queues are serviced with fixed priority: Q0 is higher than Q1. When an event is ready to be queued and the event queue and the TC channel are empty, the event bypasses the event queue and goes directly to the PaRAM processing logic for submission to the appropriate TC. If the transfer request TR bus or PaRAM processing are busy, the bypass path is not used. The bypass is not used to dequeue for a higher-priority event.

Events are extracted from the event queue when the EDMA_TPTC is available for a new TR to be programmed into the EDMA_TPTC (signaled with the empty signal, indicating an empty program register set). As an event is extracted from the event queue, the associated PaRAM entry is processed and submitted to the TPTC as a TR. The TPCC updates the appropriate counts and addresses in the PaRAM entry in anticipation of the next trigger event for that PaRAM entry.

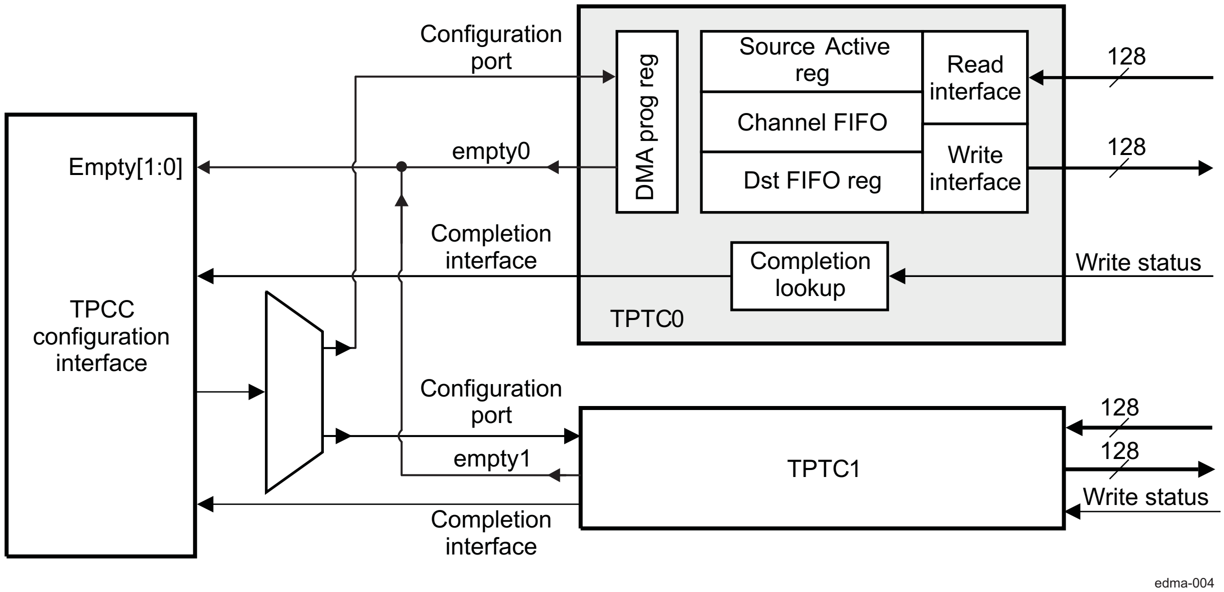
The EDMA_TPCC also has an error detection logic that causes an error interrupt generation on various error conditions (for example: missed events EDMA_TPCC_EMR and EDMA_TPCC_EMRH registers, exceeding event queue thresholds in EDMA_TPCC_CCERR register, etc.).

16.2.5.1.2 Third-Party Transfer Controller

The TPTC module is the EDMA transfer engine that generates transfers as programmed in dedicated working registers, using two dedicated master ports: a read-only port and a write-only port.

Figure 16-17 shows a functional block diagram and of the EDMA transfer controller (EDMA_TPTC) and its connection to the EDMA_TPCC.

Figure 16-17. TPTC Block Diagram



NOTE: The port data bus width of the instances of the TPTC is fixed at 128 bits.

Two instances of the EDMA_TPTC generate concurrent traffic on the L3_MAIN interconnect. Each TC controller consists of the following components:

- DMA Program Register Set: Stores the context for the DMA transfer that is loaded into the active register set when the current active register set completes. The CPU or TPCC programs the Program Register Set, not the active register set. For typical standalone operation, the CPU programs the Program Register while the TC services the Active register set. The Program Register set includes ownership control such that CPU software and the EDMA stay synchronized relative to one another.
- Source Active Register Set : Stores the context (src/dst/cnt/etc) for the DMA Transfer Request (TR) in progress in the Read Controller. The Active register set is split into independent Source and

Destination, because the source interconnect controller and the distant interconnect controller operate independently of one another.

- Destination FIFO Register Set: Stores the context (src/dst/cnt/etc) for the DMA Transfer Request (TR) in progress, or pending, in the Write Controller. The pending register must allow the source controller to begin processing a new TR while the distant register set processes the previous TR.
- Channel FIFO: Temporary holding buffer for in-flight data. The read return data of the source peripheral is stored in the Data FIFO, and then is written to the destination peripheral by the write command/data bus.
- Read Controller/Interconnect Read Interface: The Interconnect read interface issues optimally sized read commands to the source peripheral, based on a burst size of 128 bytes and available landing space in the channel FIFO.
- Write controller/Interconnect Write interface: The local interconnect write interface issues optimally sized write commands to the destination peripheral, based on a burst size of 128 bytes and available data in the channel FIFO.
- Completion interface: sends completion codes to the EDMA_TPCC when a transfer completes and generates interrupts and chained events in the TPCC module.
- Configuration port: Slave interface that provides read/write access to program registers and read access to all memory-mapped TPTC registers.

When one EDMA_TPTC module is idle and receive its first TR, DMA program register set receives the TR, where it transitions to the DMA source active set and the destination FIFO register set immediately. The second TR (if pending from EDMA_TPCC) is loaded into the DMA program set, ensuring it can start as soon as possible when the active transfer completes. 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 controlled by the rules of command fragmentation and optimization. These are issued only when the data FIFO has space available for the data read. When sufficient data is in the data FIFO, the write controller starts issuing a write command again following the rules for command fragmentation and optimization.

Depending on the number of entries, the read controller can process up to two or four transfer requests ahead of the destination subject to the amount of free data FIFO.

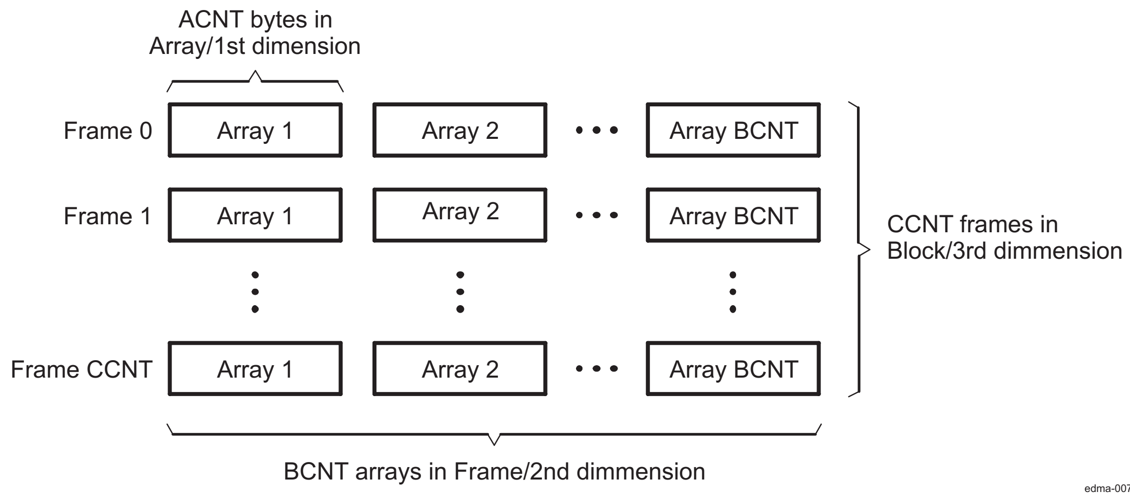
16.2.5.2 Types of EDMA controller Transfers

An EDMA transfer is always defined in terms of three dimensions. Figure 16-18 shows the three dimensions used by EDMA controller transfers. These three dimensions are defined as:

- 1st Dimension or Array (A): The 1st dimension in a transfer consists of EDMA_TPCC_ABCNT_n[15:0] ACNT contiguous bytes.
- 2nd Dimension or Frame (B): The 2nd dimension in a transfer consists of EDMA_TPCC_ABCNT_n[31:16] BCNT arrays of ACNT bytes. Each array transfer in the 2nd dimension is separated from each other by an index programmed using bit-fields EDMA_TPCC_BIDX_n[15:0] SBIDX or EDMA_TPCC_BIDX_n[31:16] DBIDX.
- 3rd Dimension or Block (C): The 3rd dimension in a transfer consists of CCNT frames of BCNT arrays of ACNT bytes. The Count for 3rd Dimension is defined in register EDMA_TPCC_CCNT_n[15:0] CCNT. Each transfer in the 3rd dimension is separated from the previous by an index programmed using EDMA_TPCC_CIDX_n[15:0] SCIDX or EDMA_TPCC_CIDX_n[31:16] DCIDX.

NOTE: 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 (EDMA_TPCC_OPT_n[2] SYNCDIM bit). For these three dimensions, only two synchronization types are supported: A-synchronized transfers and AB-synchronized transfers.

Figure 16-18. Definition of ACNT, BCNT, and CCNT



16.2.5.2.1 A-Synchronized Transfers

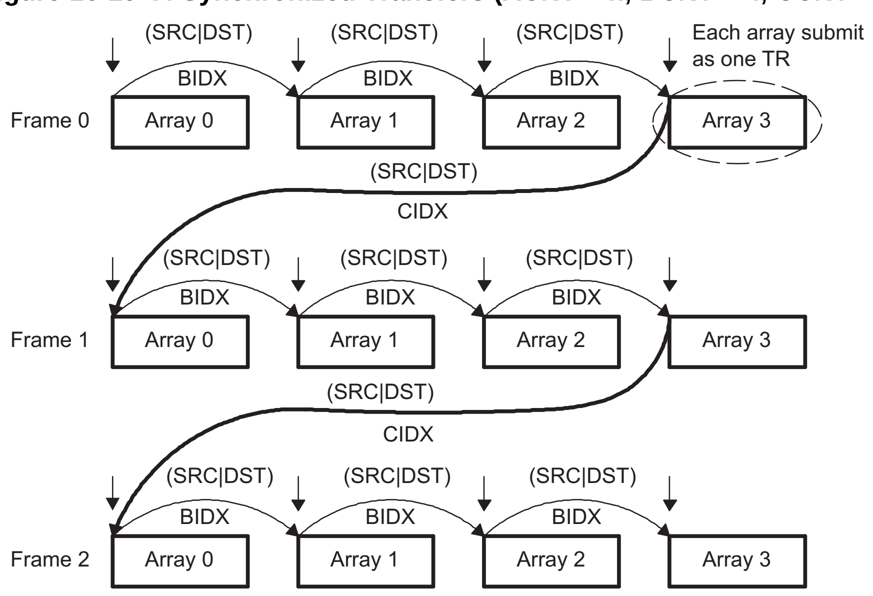
In an A-synchronized transfer, each EDMA sync event initiates the transfer of the 1st dimension of EDMA_TPCC_ABCNT_n[15:0] ACNT bytes, or one array of ACNT bytes. Each event/TR packet conveys the transfer information for one array only. Thus, BCNT × CCNT events are needed to completely service a PaRAM set.

Arrays are always separated by EDMA_TPCC_BIDX_n[15:0]SBIDX and EDMA_TPCC_BIDX_n[31:16] DBIDX, as shown in Figure 16-19, where the start address of Array N is equal to the start address of Array N – 1 plus source (SRC) or destination (DST) in EDMA_TPCC_BIDX_n register.

Frames are always separated by EDMA_TPCC_CIDX_n[15:0] SCIDX and EDMA_TPCC_CIDX_n[31:16] DCIDX. 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 16-19, SRCCIDX / DSTCIDX is the difference between the start of Frame 0 Array 3 to the start of Frame 1 Array 0.

Figure 16-19 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 Figure 16-19 for details on parameter set updates.

Figure 16-19. A-Synchronized Transfers (ACNT = n, BCNT = 4, CCNT = 3)



16.2.5.2.2 AB-Synchronized Transfers

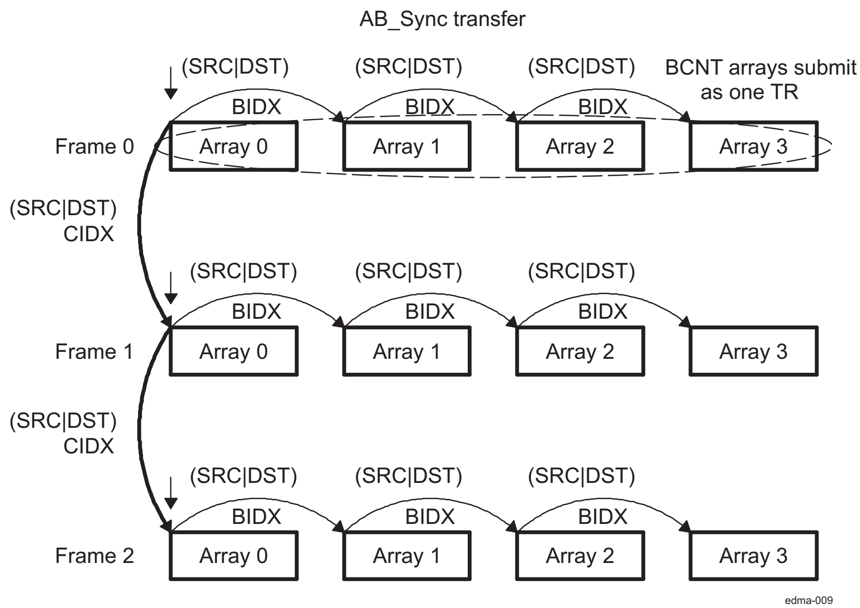
In a AB-synchronized transfer, each EDMA sync event initiates the transfer of 2 dimensions or one frame. Each event/TR packet conveys information for one entire frame of BCNT_n arrays of ACNT_n bytes. Thus, EDMA_TPCC_CCNT_n events are needed to completely service a PaRAM set.

Arrays are always separated by EDMA_TPCC_BIDX_n[15:0] SBIDX and EDMA_TPCC_BIDX_n[31:16] DBIDX as shown in Figure 16-20. 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 EDMA_TPCC_CIDX_n[15:0] SCIDX / EDMA_TPCC_CIDX_n[31:16] DCIDX 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 16.2.5.3.6 *Parameter Set Updates* for details on parameter set updates.

Figure 16-20 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 16-20. AB-Synchronized Transfers (ACNT = n, BCNT = 4, CCNT = 3)



NOTE: ABC-synchronized transfers are not directly supported. It can be logically achieved by chaining between multiple AB-synchronized transfers.

NOTE: VCP does not support Const/FIFO mode DMA transfers. The EDMA should be configured for AB-Synchronized transfer with ACNT = 8, BCNT = number of elements.

16.2.5.3 Parameter RAM (PaRAM)

The EDMA 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 in EDMA_TPCC. The PaRAM table is segmented into multiple PaRAM sets. Each PaRAM set includes eight four-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.

The PaRAM structure supports flexible ping-pong, circular buffering, channel chaining, and auto-reloading (linking).

The contents of the PaRAM include the following:

- 512 PaRAM sets
- 64 channels that are direct mapped and can be used as link or QDMA sets if not used for DMA channels
- 64 channels remain for link or QDMA sets

By default, all channels map to PaRAM set to 0, they should be remapped before use by EDMA_TPCC_DCHMAPN_m and EDMA_TPCC_QCHMAPN_j registers.

Table 16-90. EDMA Parameter RAM Contents

PaRAM Set Number	Base Address	Parameters (1)
0	EDMA Base Address + 4000h to EDMA Base Address + 401Fh	PaRAM set 0
1	EDMA Base Address + 4020h to EDMA Base Address + 403Fh	PaRAM set 1
2	EDMA Base Address + 4040h to EDMA Base Address + 405Fh	PaRAM set 2
3	EDMA Base Address + 4060h to EDMA Base Address + 407Fh	PaRAM set 3
4	EDMA Base Address + 4080h to EDMA Base Address + 409Fh	PaRAM set 4
5	EDMA Base Address + 40A0h to EDMA Base Address + 40BFh	PaRAM set 5
6	EDMA Base Address + 40C0h to EDMA Base Address + 40DFh	PaRAM set 6
7	EDMA Base Address + 40E0h to EDMA Base Address + 40FFh	PaRAM set 7
8	EDMA Base Address + 4100h to EDMA Base Address + 411Fh	PaRAM set 8
9	EDMA Base Address + 4120h to EDMA Base Address + 413Fh	PaRAM set 9
...	...	...
63	EDMA Base Address + 47E0h to EDMA Base Address + 47FFh	PaRAM set 63
64	EDMA Base Address + 4800h to EDMA Base Address + 481Fh	PaRAM set 64
65	EDMA Base Address + 4820h to EDMA Base Address + 483Fh	PaRAM set 65
...	...	...
510	EDMA Base Address + 7FC0h to EDMA Base Address + 7FDFh	PaRAM set 510
511	EDMA Base Address + 7FE0h to EDMA Base Address + 7FFFh	PaRAM set 511

(1) The device has 8 QDMA channels that can be mapped to any parameter set number from 0 to 511.

16.2.5.3.1 PaRAM

Each parameter set of PaRAM is organized into eight 32-bit words or 32 bytes, as shown in Figure 16-21 and described in Table 16-91. Each PaRAM set consists of 16-bit and 32-bit parameters.

Figure 16-21. PaRAM Set

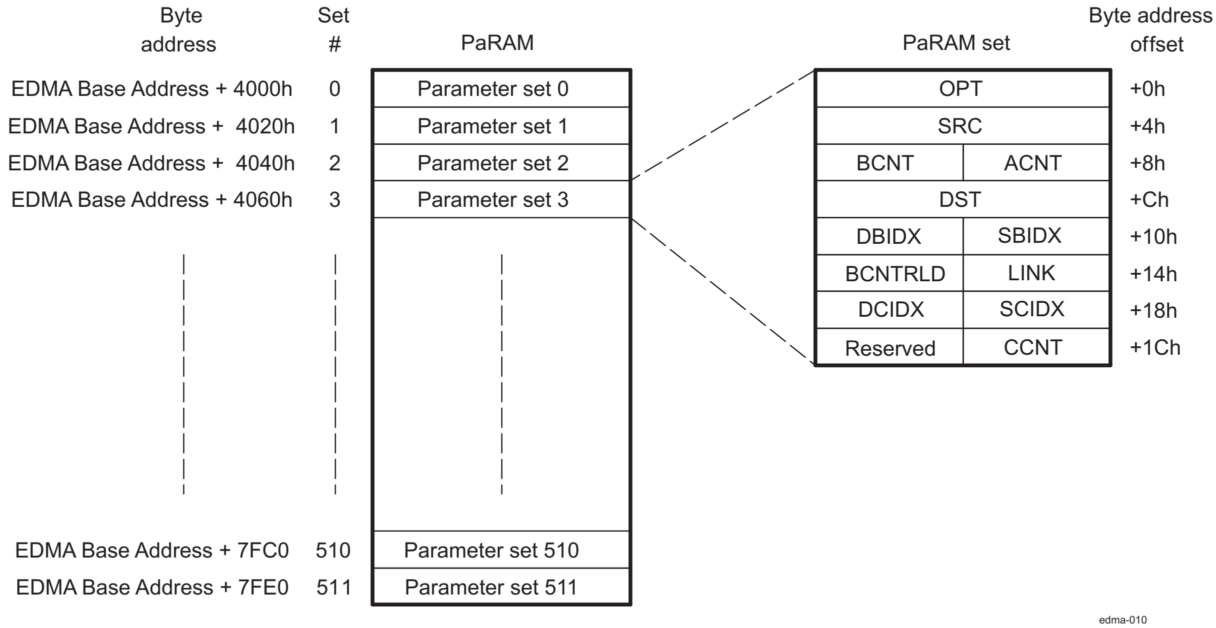


Table 16-91. EDMA Channel Parameter Description

Offset Address (bytes)	Acronym	Parameter	Description
0h	OPT	Channel Options EDMA_TPCC_OPT_n register	Transfer configuration options
4h	SRC	Channel Source Address EDMA_TPCC_SRC_n register	The byte address from which data is transferred
8h[(1)]	ACNT	Count for 1st Dimension EDMA_TPCC_ABCNT_n[15:0] ACNT bit-field.	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 EDMA_TPCC_ABCNT_n[31:16] BCNT bit-field.	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 EDMA_TPCC_DST_n register	The byte address to which data is transferred
10h[(1)]	SBIDX	Source BCNT Index EDMA_TPCC_BIDX_n[15:0] SBIDX bit-field.	Signed value specifying the byte address offset between source arrays within a frame (2nd dimension). Valid values range from –32 768 and 32 767.
	DBIDX	Destination BCNT Index EDMA_TPCC_BIDX_n[31:16] DBIDX bit-field.	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 EDMA_TPCC_LNK_n[15:0] LINK bit-field	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 EDMA_TPCC_LNK_n[31:16] BCNTRLD bit-field	The count value used to reload BCNT when BCNT decrements to 0 (TR is submitted for the last array in 2nd dimension). Only relevant in A-synchronized transfers.
18h[(1)]	SCIDX	Source CCNT index. EDMA_TPCC_CIDX_n[15:0] SCIDX bit-field.	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.
	DCIDX	Destination CCNT index. EDMA_TPCC_CIDX_n[31:16] DCIDX bit-field.	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. EDMA_TPCC_CCNT_n[15:0] CCNT bit-field.	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.
	Reserved	Reserved	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.

(1) If OPT, SRC, or DST is the trigger word for a QDMA transfer, then a 32-bit access to that field is required. 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 will swap addresses of the 16-bit fields, but 32-bit accesses avoid the issue entirely.

16.2.5.3.2 EDMA Channel PaRAM Set Entry Fields

16.2.5.3.2.1 Channel Options Parameter (OPT)

For detailed information about the channel options parameter, see the EDMA_TPCC_OPT_n register description in Section 16.2.8.2.2.1, *EDMA_TPCC Register Description*.

16.2.5.3.2.2 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 EDMA. For SAM in constant addressing mode, it must program the source address to be aligned to a 256-bit aligned address (5 LSBs of address must be 0). If this rule is not observed, the EDMA_TPTC returns an error. Refer to Section 16.2.5.12.3 *Error Generation* for additional details.

16.2.5.3.2.3 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 EDMA. For DAM in constant addressing mode, it must program the destination address to be aligned to a 256-bit aligned address (5 LSBs of address must be 0). If this rule is not observed, the EDMA_TPTC returns an error. Refer to Section 16.2.5.12.3 *Error Generation* for additional details.

16.2.5.3.2.4 Count for 1st Dimension (ACNT)

EDMA_TPCC_ABCNT_n[15:0] 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 EDMA_TPTC. A transfer with ACNT equal to 0 is considered either a null or dummy transfer. A dummy or null transfer generates a completion code depending on the settings of the completion bit fields in EDMA_TPCC_OPT_n.

Refer to Section 16.2.5.3.5 *Dummy Versus Null Transfer Comparison* and Section 16.2.5.5.3 *Dummy or Null Completion* for details on dummy/null completion conditions.

16.2.5.3.2.5 Count for 2nd Dimension (BCNT)

EDMA_TPCC_ABCNT_n[15:0] 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. A dummy or null transfer generates a completion code depending on the settings of the completion bit fields in EDMA_TPCC_OPT_n.

Refer to Section 16.2.5.3.5 *Dummy Versus Null Transfer Comparison* and Section 16.2.5.5.3 *Dummy or Null Completion* for details on dummy/null completion conditions.

16.2.5.3.2.6 Count for 3rd Dimension (CCNT)

EDMA_TPCC_CCNT_n[15:0] 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. A dummy or null transfer generates a completion code depending on the settings of the completion bit fields in EDMA_TPCC_OPT_n.

A CCNT value of 0 is considered either a null or dummy transfer.

Refer to Section 16.2.5.3.5 *Dummy Versus Null Transfer Comparison* and Section 16.2.5.5.3 *Dummy or Null Completion* for details on dummy/null completion conditions.

16.2.5.3.2.7 BCNT Reload (BCNTRLD)

EDMA_TPCC_LNK_n[31:16] BCNTRLD is a 16-bit unsigned value used to reload the EDMA_TPCC_ABCNT_n[15:0] 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 EDMA_TPCC decrements the BCNT value by 1 on each TR submission. When BCNT reaches 0, the EDMA_TPCC decrements CCNT and uses the BCNTRLD value to reinitialize the BCNT value.

For AB-synchronized transfers, the EDMA_TPCC submits the BCNT in the TR and the EDMA_TPTC decrements BCNT appropriately. For AB-synchronized transfers, BCNTRLD is not used.

16.2.5.3.2.8 Source B Index (SBIDX)

EDMA_TPCC_BIDX_n[15:0] SBIDX is a 16-bit signed value (2s complement) used for source address modification between each array in the 2nd dimension. Valid values for EDMA_TPCC_BIDX_n[15:0] SBIDX 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:

- EDMA_TPCC_BIDX_n[15:0] SBIDX = 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.
- EDMA_TPCC_BIDX_n[15:0] SBIDX = 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.
- EDMA_TPCC_BIDX_n[15:0] SBIDX = 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.

16.2.5.3.2.9 Destination B Index (DBIDX)

EDMA_TPCC_BIDX_n[31:16] DBIDX is a 16-bit signed value (2s complement) used for destination address modification between each array in the 2nd dimension. Valid values for EDMA_TPCC_BIDX_n[31:16] DBIDX 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. Refer to Section 16.2.5.3.2.8 *Source B Index (SBIDX)* for examples.

16.2.5.3.2.10 Source C Index (SCIDX)

EDMA_TPCC_CIDX_n[15:0] SCIDX is a 16-bit signed value (2s complement) used for source address modification in the 3rd dimension. Valid values for EDMA_TPCC_CIDX_n[15:0] SCIDX 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: When SCIDX is applied, the current array in an A-synchronized transfer is the last array in the frame (Figure 16-19), while the current array in an AB-synchronized transfer is the first array in the frame (Figure 16-20).

16.2.5.3.2.11 Destination C Index (DCIDX)

EDMA_TPCC_CIDX_n[31:16] DCIDX 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: When DCIDX is applied, the current array in an A-synchronized transfer is the last array in the frame (Figure 16-19), while the current array in a AB-synchronized transfer is the first array in the frame (Figure 16-20).

16.2.5.3.2.12 Link Address (LINK)

The EDMA_TPCC 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 EDMA_TPCC_LNK_n[15:0] LINK specifies the byte address offset in the PaRAM from which the EDMA_TPCC loads/reloads the next PaRAM set during linking.

It 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 EDMA_TPCC ignores the upper 2 bits of the LINK entry, allowing the flexibility of programming the link address as either an absolute/literal byte address or use the PaRAM-base-relative offset address. Therefore, if it use the literal address with a range from 4000h to 7FFFh, it will be treated as a PaRAM-base-relative value of 0000h to 3FFFh.

It should check that the programed value in the EDMA_TPCC_LNK_n[15:0] LINK field is correctly, so that link update is requested from a PaRAM address that falls in the range of the available PaRAM addresses on the device.

Value of FFFFh in EDMA_TPCC_LNK_n[15:0] LINK bit-field is referred to as a NULL link that should cause the EDMA_TPCC to perform an internal write of 0 to all entries of the current PaRAM set, except for the EDMA_TPCC_LNK_n[15:0] LINK field is set to FFFFh. Also, see Section 16.2.5.5 *Completion of a DMA Transfer* for details on terminating a transfer.

16.2.5.3.3 Null PaRAM Set

A null PaRAM set is defined as a PaRAM set where all count fields (EDMA_TPCC_ABCNT_n[15:0] ACNT, EDMA_TPCC_ABCNT_n[31:16] BCNT, and EDMA_TPCC_CCNT_n[15:0] CCNT) are cleared to 0. If a PaRAM set associated with a channel is a NULL set, then when serviced by the EDMA_TPCC, the bit corresponding to the channel is set in the associated event missed register (EDMA_TPCC_EMR, EDMA_TPCC_EMRH, or EDMA_TPCC_QEMR). This bit remains set in the associated secondary event register (EDMA_TPCC_SER, EDMA_TPCC_SERH, or EDMA_TPCC_QSER).
This implies that any future events on the same channel are ignored by the EDMA_TPCC and it is required to clear the bit in EDMA_TPCC_SER, EDMA_TPCC_SERH, or EDMA_TPCC_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.

16.2.5.3.4 Dummy PaRAM Set

A dummy PaRAM set is defined as a PaRAM set where at least one of the count fields (EDMA_TPCC_ABCNT_n[15:0] ACNT, EDMA_TPCC_ABCNT_n[31:16] BCNT, or EDMA_TPCC_CCNT_n[15:0] 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 EDMA_TPCC, it will not set the bit corresponding to the channel (DMA/QDMA) in the event missed register (EDMA_TPCC_EMR, EDMA_TPCC_EMRH, or EDMA_TPCC_QEMR) and the secondary event register (EDMA_TPCC_SER, EDMA_TPCC_SERH, or EDMA_TPCC_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.

16.2.5.3.5 Dummy Versus Null Transfer Comparison

There are some differences in the way the EDMA_TPCC 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 EDMA_TPCC_EMR to get set and the E*n* bit in EDMA_TPCC_SER remains set, essentially preventing any further transfers on that channel without clearing the associated error registers.

Table 16-92 summarizes the conditions and effects of null and dummy transfer requests.

Table 16-92. Dummy and Null Transfer Request

Feature	Null TR	Dummy TR
EDMA_TPCC_EMR / EDMA_TPCC_EMRH / EDMA_TPCC_QEMR is set	Yes	No
EDMA_TPCC_SER / EDMA_TPCC_SERH / EDMA_TPCC_QSER remains set	Yes	No
Link update (STATIC = 0 in EDMA_TPCC_OPT_n)	Yes	Yes
EDMA_TPCC_QER is set	Yes	Yes
EDMA_TPCC_IPR / EDMA_TPCC_IPRH, EDMA_TPCC_CER / EDMA_TPCC_CERH is set using early completion	Yes	Yes

16.2.5.3.6 Parameter Set Updates

When a TR is submitted for a given DMA/QDMA channel and its corresponding PaRAM set, the EDMA_TPCC is responsible for updating the PaRAM set in anticipation of the next trigger event. For events that are not final, 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 EDMA_TPCC_ABCNT_n[31:16] 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 EDMA_TPCC_ABCNT_n[15:0] ACNT byte transfers have submitted. For AB-synchronized transfers, a C-update refers to the decrementing of EDMA_TPCC_CCNT_n[15:0] CCNT after submission of every transfer request.

Refer to Table 16-93 for details and conditions on the parameter updates. A link update occurs when the PaRAM set is exhausted, as described in Section 16.2.5.3.7 *Linking Transfers*.

After the TR is read from the PaRAM (and is in process of being submitted to EDMA_TPTC), 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: EDMA_TPCC_ABCNT_n[15:0] ACNT, EDMA_TPCC_LNK_n[31:16] BCNTRLD, EDMA_TPCC_BIDX_n[15:0] SBIDX, EDMA_TPCC_BIDX_n[31:16] DBIDX, EDMA_TPCC_CIDX_n[15:0] SCIDX, EDMA_TPCC_CIDX_n[31:16] DCIDX, EDMA_TPCC_OPT_n, EDMA_TPCC_LNK_n[15:0]LINK.
- AB-synchronized: EDMA_TPCC_ABCNT_n[15:0] ACNT, EDMA_TPCC_ABCNT_n[31:16] BCNT, EDMA_TPCC_LNK_n[31:16] BCNTRLD, EDMA_TPCC_BIDX_n[15:0] SBIDX, EDMA_TPCC_BIDX_n[31:16] DBIDX, EDMA_TPCC_CIDX_n[15:0] SCIDX, EDMA_TPCC_CIDX_n[31:16] DCIDX, EDMA_TPCC_OPT_n, EDMA_TPCC_LNK_n[15:0]LINK.

NOTE: PaRAM updates only pertain to the information that is needed to properly submit the next transfer request to the EDMA_TPTC. Updates that occur while data is moved within a transfer request are tracked within the transfer controller, and is detailed in Section 16.2.5.12 *EDMA Transfer Controller (EDMA_TPTC)*. For A-synchronized transfers, the EDMA_TPCC always submits a TRP for EDMA_TPCC_ABCNT_n[15:0] ACNT bytes (EDMA_TPCC_ABCNT_n[31:16] BCNT = 1 and EDMA_TPCC_CCNT_n[15:0] CCNT = 1). For AB-synchronized transfers, the EDMA_TPCC always submits a TRP for EDMA_TPCC_ABCNT_n[15:0] ACNT bytes of BCNT arrays (EDMA_TPCC_CCNT_n[15:0] CCNT = 1). The EDMA_TPTC is responsible for updating source and destination addresses within the array based on EDMA_TPCC_ABCNT_n[15:0] ACNT and EDMA_TPCC_OPT_n[10:8] FWID. For AB-synchronized transfers, the EDMA_TPTC is also responsible to update source and destination addresses between arrays based on EDMA_TPCC_BIDX_n[15:0] SBIDX and EDMA_TPCC_BIDX_n[31:16] DBIDX.

Table 16-93 shows the details of parameter updates that occur within EDMA_TPCC for A-synchronized and AB-synchronized transfers.

Table 16-93. Parameter Updates in EDMA_TPCC (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**	**EDMA_TPCC_CCNT_n[15:0] CCNT > 1**	**EDMA_TPCC_CCNT_n[15:0] CCNT == 1**
SRC	+= SBIDX	+= SCIDX	= Link.EDMA_TPCC_SRC_n	in EDMA_TPTC	+= SCIDX	= Link.EDMA_TPCC_SRC_n
DST	+= DBIDX	+= DCIDX	= Link.EDMA_TPCC_DST_n	in EDMA_TPTC	+= DCIDX	= Link.EDMA_TPCC_DST_n
ACNT	None	None	= Link.EDMA_TPCC_ABCNT_n[15:0] ACNT	None	None	= Link.EDMA_TPCC_ABCNT_n[15:0] ACNT
BCNT	–= 1	= BCNTRLD	= Link.EDMA_TPCC_ABCNT_n[31:16] BCNT	in EDMA_TPTC	N/A	= Link.EDMA_TPCC_ABCNT_n[31:16] BCNT
CCNT	None	–= 1	= Link.EDMA_TPCC_CCNT_n[15:0] CCNT	in EDMA_TPTC	-=1	= Link.EDMA_TPCC_CCNT_n[15:0] CCNT
SBIDX	None	None	= Link.EDMA_TPCC_BIDX_n[15:0] SBIDX	in EDMA_TPTC	None	= Link.EDMA_TPCC_BIDX_n[15:0] SBIDX
DBIDX	None	None	= Link.EDMA_TPCC_BIDX_n[31:16] DBIDX	None	None	= Link.EDMA_TPCC_BIDX_n[31:16] DBIDX
SCIDX	None	None	= Link.EDMA_TPCC_BIDX_n[15:0] SBIDX	in EDMA_TPTC	None	= Link.EDMA_TPCC_BIDX_n[15:0] SBIDX
DCIDX	None	None	= Link.EDMA_TPCC_BIDX_n[31:16] DBIDX	None	None	= Link.EDMA_TPCC_BIDX_n[31:16] DBIDX
LINK	None	None	= Link.EDMA_TPCC_LNK_n[15:0]LINK	None	None	= Link.EDMA_TPCC_LNK_n[15:0]LINK
BCNTRLD	None	None	= Link.EDMA_TPCC_LNK_n[31:16] BCNTRLD	None	None	= Link.EDMA_TPCC_LNK_n[31:16] BCNTRLD
OPT(1)	None	None	= LINK.EDMA_TPCC_OPT_n	None	None	= LINK.EDMA_TPCC_OPT_n

(1) In all cases, no updates occur if EDMA_TPCC_OPT_n[3] STATIC == 1 for the current PaRAM set.

NOTE: The EDMA_TPCC 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. It 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.

16.2.5.3.7 Linking Transfers

The EDMA_TPCC 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 with no CPU intervention. Upon completion of a transfer, the current transfer parameters are reloaded with the parameter set pointed that the 16-bit link address field of the current parameter set points to. Linking only occurs when the EDMA_TPCC_OPT_n[3] STATIC bit is cleared.

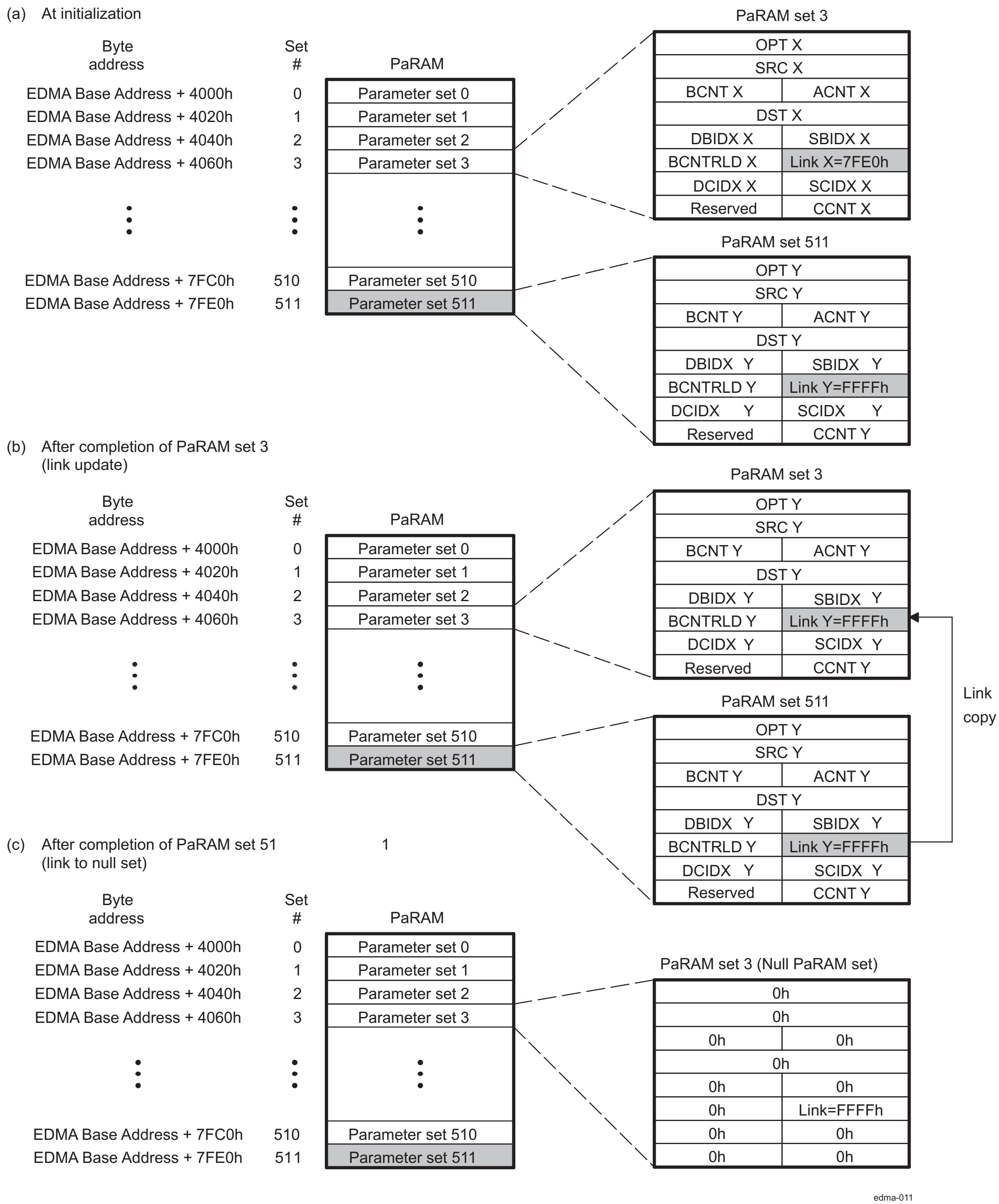
NOTE: It should always link a transfer (EDMA or QDMA) to another useful transfer. If it must terminate a transfer, then link the transfer to a NULL parameter set. Refer to Section 16.2.5.3.3 *Null PaRAM Set*.

The link update occurs after the current PaRAM set event parameters have been exhausted. An event's parameters are exhausted when the EDMA channel controller has submitted all of the transfers that are associated with the PaRAM set.

A link update occurs for null and dummy transfers depending on the state of the EDMA_TPCC_OPT_n[3] STATIC bit and the EDMA_TPCC_LNK_n[15:0] LINK field. In both cases (null or dummy), if the value of EDMA_TPCC_LNK_n[15:0] LINK is FFFFh, then a null PaRAM set (with all 0s and EDMA_TPCC_LNK_n[15:0] LINK set to FFFFh) is written to the current PaRAM set. Similarly, if EDMA_TPCC_LNK_n[15:0] LINK is set to a value other than FFFFh, then the appropriate PaRAM location that EDMA_TPCC_LNK_n[15:0] LINK points to is copied to the current PaRAM set.

Once the channel completion conditions are met for an event, the transfer parameters that are located at the link address are loaded into the current DMA or QDMA channel's associated parameter set. This indicates that the EDMA_TPCC reads the entire set (eight words) from the PaRAM set specified by EDMA_TPCC_LNK_n[15:0] LINK and writes all eight words to the PaRAM set that is associated with the current channel. Figure 16-22 shows an example of a linked transfer.

Figure 16-22. Linked Transfer

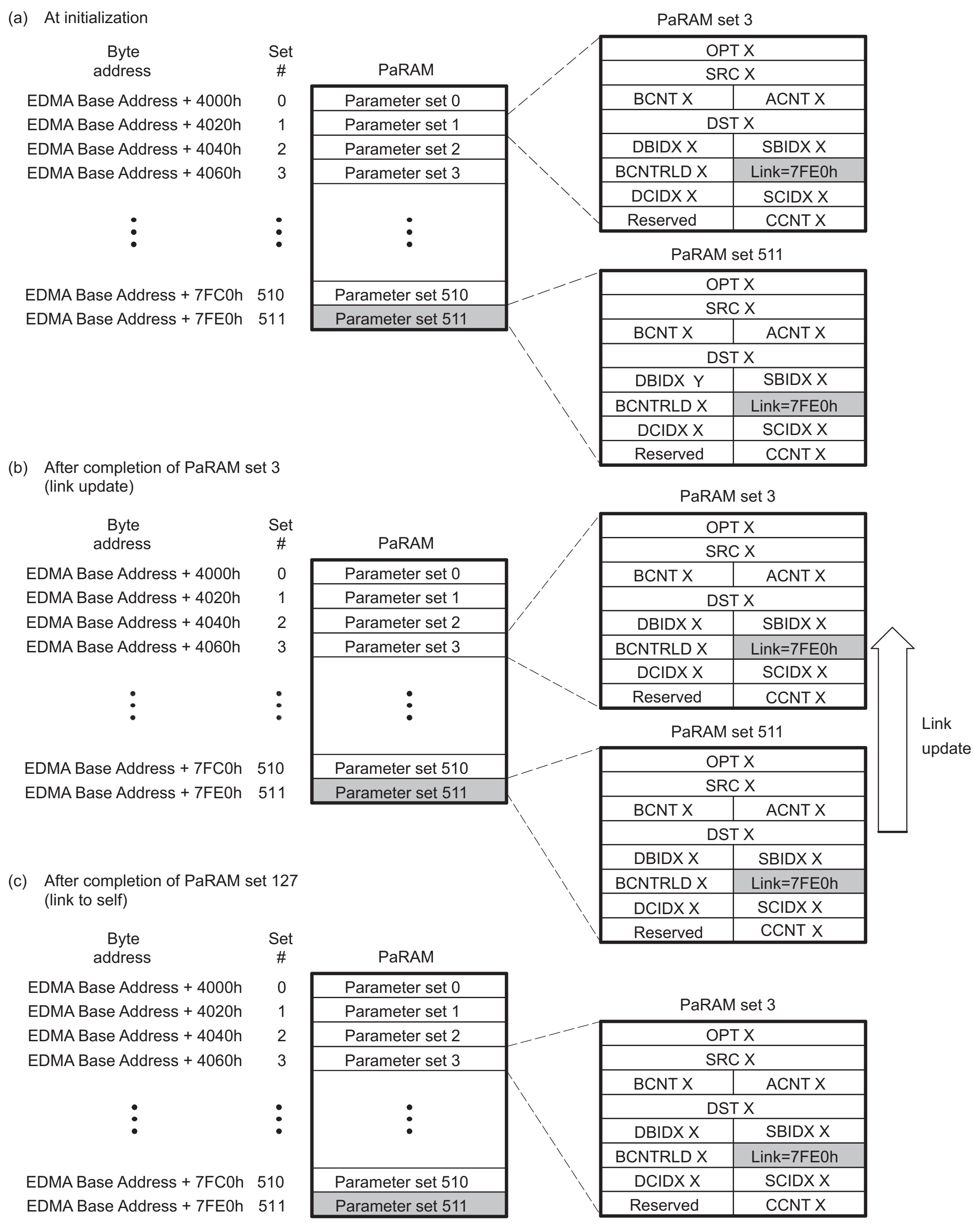


Any PaRAM set in the PaRAM can be used as a link/reload parameter set. The PaRAM sets associated with peripheral synchronization events (refer to Section 16.2.5.6 *Event, Channel, and PaRAM Mapping*) only use for linking if the corresponding events are disabled.

If a PaRAM set location is defined as a QDMA channel PaRAM set (by EDMA_TPCC_QCHMAPN_j register), then copying the link PaRAM set into the current QDMA channel PaRAM set is recognized as a trigger event. It is latched in EDMA_TPCC_QER because a write to the trigger word was performed. This feature is used to create a linked list of transfers using a single QDMA channel and multiple PaRAM sets. Refer to Section 16.2.5.4.2 *QDMA Channels*.

Linking to itself replicates the behavior of auto-initialization, thus facilitating the use of circular buffering and repetitive transfers. After an EDMA channel exhausts its current PaRAM set, it reloads all of the parameter set entries from another PaRAM set, which is initialized with values that are identical to the original PaRAM set. Figure 16-23 shows an example of a linked to self transfer. Here, the PaRAM set 511 has the link field pointing to the address of parameter set 511 (linked to self).

Figure 16-23. Link-to-Self Transfer



NOTE: If the in EDMA_TPCC_OPT_n[3] STATIC bit is set for a PaRAM set, then link updates are not performed.

16.2.5.3.8 Constant Addressing Mode Transfers/Alignment Issues

If either EDMA_TPCC_OPT_n[0] SAM or EDMA_TPCC_OPT_n[1] DAM is set (constant addressing mode), then the source or destination address must be aligned to a 256-bit aligned address, respectively, and the corresponding EDMA_TPCC_BIDX_n is an even multiple of 32 bytes (256 bits). The EDMA_TPCC does not recognize errors here, but the EDMA_TPTC asserts an error if this is not true. Refer to Section 16.2.5.12.3 *Error Generation*.

> **NOTE:** The constant addressing (CONST) mode has limited applicability. The EDMA is configured for the constant addressing mode (EDMA_TPCC_OPT_n[0] SAM / EDMA_TPCC_OPT_n[1] DAM = 1) only if the transfer source or destination (on-chip memory, off-chip memory controllers, slave peripherals) support the constant addressing mode. If the constant addressing mode is not supported, the similar logical transfer can be achieved using the increment (INCR) mode (EDMA_TPCC_OPT_n[0] SAM / EDMA_TPCC_OPT_n[1] DAM =0) by appropriately programming the count and indices values.

16.2.5.3.9 Element Size

The EDMA controller does not use element-size and element-indexing. Instead, all transfers are defined in terms of all three dimensions: EDMA_TPCC_ABCNT_n[15:0] ACNT, EDMA_TPCC_ABCNT_n[31:16] BCNT, and EDMA_TPCC_CCNT_n[15:0] CCNT. An element-indexed transfer is logically achieved by programming EDMA_TPCC_ABCNT_n[15:0] ACNT to the size of the element and EDMA_TPCC_ABCNT_n[31:16] BCNT to the number of elements that need to be transferred. For example: If there are 16-bit audio data and 256 audio samples that must be transferred to a serial port, therefore the EDMA_TPCC_ABCNT_n[15:0] ACNT = 2 (2 bytes) and EDMA_TPCC_ABCNT_n[31:16] BCNT = 256.

16.2.5.4 Initiating a DMA Transfer

There are multiple ways to initiate a programmed data transfer using the EDMA_TPCC channel controller. Transfers on DMA channels are initiated by three sources.

They are listed as follows:

- **Event-triggered transfer request** (this is the typical usage of EDMA controller): A peripheral, system, or externally-generated event triggers 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 registers (EDMA_TPCC_ESR / EDMA_TPCC_ESRH).
- **Chain-triggered transfer request:** A transfer is triggered on the completion of another transfer or sub-transfer.

Transfers on QDMA channels are initiated by two sources. They are as follows:

- **Auto-triggered transfer request:** Writing to the programmed trigger word triggers a transfer.
- **Link-triggered transfer requests:** Writing to the trigger word triggers the transfer when linking occurs.

16.2.5.4.1 DMA Channel

16.2.5.4.1.1 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 (EDMA_TPCC_ER[31:0] E*n* = 1). For more information about peripheral events to EDMA events mapping, refer to *the device data manual*.
If the corresponding event in the event enable register (EDMA_TPCC_EER) is enabled (EDMA_TPCC_EER[31:0] E*n* = 1), then the EDMA_TPCC 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 EDMA_TPTC and the EDMA_TPCC_ER[31:0] E*n* bit is cleared. At this point, a new event can be safely received by the EDMA_TPCC.

If the PaRAM set associated with the channel is a NULL set (see Section 16.2.5.3.3 *Null PaRAM Set*), then no transfer request (TR) is submitted and the corresponding EDMA_TPCC_ER[31:0] E*n* bit is cleared and simultaneously the corresponding channel bit is set in the event miss register (EDMA_TPCC_EMR[31:0] 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 re-triggering the DMA channel.

When an event is received, the corresponding event bit in the event register is set (EDMA_TPCC_ER[31:0] E*n* = 1), regardless of the state of EDMA_TPCC_EER[31:0] E*n*. If the event is disabled when an external event is received (EDMA_TPCC_ER[31:0] E*n* = 1 and EDMA_TPCC_EER[31:0] E*n* = 0), the EDMA_TPCC_ER[31:0] E*n* bit remains set. If the event is subsequently enabled (EDMA_TPCC_EER[31:0] E*n* = 1), then the pending event is processed by the EDMA_TPCC and the TR is processed/submitted, after which the EDMA_TPCC_ER[31:0] 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 (EDMA_TPCC_ER[31:0] E*n* != 0), then the second event is registered as a missed event in the corresponding bit of the event missed register (EDMA_TPCC_EMR[31:0] E*n* = 1).

16.2.5.4.1.2 Manually-Triggered Transfer Request

The CPU or any peripheral device module initiates a DMA transfer by writing to the event set register EDMA_TPCC_ESR. Writing a 1 to an event bit in the EDMA_TPCC_ESR results in the event being prioritized/queued in the appropriate event queue, regardless of the state of the EDMA_TPCC_EER[31:0] 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 EDMA_TPTC and the channel can be triggered again.

If the PaRAM set associated with the channel is a NULL set (see Section 16.2.5.3.3 *Null PaRAM Set*), then no transfer request (TR) is submitted and the corresponding EDMA_TPCC_ER[31:0] E*n* bit is cleared and simultaneously the corresponding channel bit is set in the event miss register EDMA_TPCC_EMR[31:0] 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 re-triggering 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 EDMA_TPCC_ESR[31:0] E*n* = 1 prior to the original being cleared EDMA_TPCC_ESR[31:0] E*n* = 0, then the second event is registered as a missed event in the corresponding bit of the event missed register EDMA_TPCC_EMR[31:0] E*n* = 1.

16.2.5.4.1.3 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 EDMA_TPCC_OPT_n[17:12] TCC of the PaRAM set associated with the channel, it results in the corresponding bit in the chained event register EDMA_TPCC_CER to be set EDMA_TPCC_CER[31:0] E[TCC] = 1).

Once a bit is set in EDMA_TPCC_CER, the EDMA_TPCC 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 EDMA_TPTC and the channel can be triggered again.

If the PaRAM set associated with the channel is a NULL set (see Section 16.2.5.3.3 *Null PaRAM Set*), then no transfer request (TR) is submitted and the corresponding EDMA_TPCC_CER[31:0] E*n* bit is cleared and simultaneously the corresponding channel bit is set in the event miss register EDMA_TPCC_EMR[31:0] 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 before the DMA channel can be re-triggered. Good programming practices might include clearing the event missed error before re-triggering 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 EDMA_TPCC_CER[31:0] E*n* != 0), then the second chained event is registered as a missed event in the corresponding channel bit of the event missed register EDMA_TPCC_EMR[31:0] E*n* = 1.

NOTE: Chained event registers EDMA_TPCC_CER, event registers EDMA_TPCC_ER, and event set registers EDMA_TPCC_ESR operate independently. An event E*n* can be triggered by any of the trigger sources (event-triggered, manually-triggered, or chain-triggered).

16.2.5.4.2 QDMA Channels

16.2.5.4.2.1 Auto-triggered and Link-Triggered Transfer Request

QDMA-based transfer requests are issued when a QDMA event gets latched in the QDMA event register EDMA_TPCC_QER[31:0] E*n* = 1. A bit corresponding to a QDMA channel is set in the QDMA event register EDMA_TPCC_QER when the following occurs:

- A CPU (or any device module) write occurs to a PaRAM address that is defined as a QDMA channel trigger word (programmed in the QDMA channel mapping register EDMA_TPCC_QCHMAPN_j for the particular QDMA channel and the QDMA channel is enabled via the QDMA event enable register EDMA_TPCC_QEER[31:0] E*n* = 1.
- EDMA_TPCC performs a link update on a PaRAM set address that is configured as a QDMA channel matches EDMA_TPCC_QCHMAPN_j settings and the corresponding channel is enabled via the QDMA event enable register EDMA_TPCC_QEER[31:0] E*n* = 1.

Once a bit is set in EDMA_TPCC_QER, the EDMA_TPCC 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 EDMA_TPTC and the channel can be triggered again.

If a bit is already set in EDMA_TPCC_QER[31:0] 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 EDMA_TPCC_QEMR[7:0] E*n* = 1.

16.2.5.4.3 Comparison Between DMA and QDMA Channels

The primary difference between DMA and QDMA channels is the event/channel synchronization.

QDMA events are either auto-triggered or link triggered. Auto-triggering 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 device modules) writes to the trigger word of the QDMA channel parameter set (auto-triggered) or when the EDMA_TPCC performs a link update on a PaRAM set that has been mapped to a QDMA channel (link triggered).

NOTE: The CPUs triggered (manually triggered) DMA channels, in addition to writing to the PaRAM set, it is required to write to the event set register EDMA_TPCC_ESR to kick-off the transfer.

QDMA channels are typically for cases where a single event accomplishes a complete transfer since the CPU (or other device modules) must reprogram some portion of the QDMA PaRAM set in order to re-trigger the channel. QDMA transfers are programmed with EDMA_TPCC_ABCNT_n[31:0] BCNT =1 and EDMA_TPCC_CCNT_n[15:0] CCNT = 1 for A-synchronized transfers, and EDMA_TPCC_CCNT_n[15:0] CCNT = 1 for AB-synchronized transfers.

Additionally, since linking is also supported (if EDMA_TPCC_OPT_n[3] STATIC = 0) for QDMA transfers, it allows to initiate a linked list of QDMAs, so when EDMA_TPCC copies over a link PaRAM set (including the write to the trigger word), the current PaRAM set mapped to the QDMA channel automatically recognizes as a valid QDMA event and initiate another set of transfers as specified by the linked set.

16.2.5.5 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 16-94 for both synchronization types along with state of the PaRAM set prior to the final TR being submitted. When the counts (EDMA_TPCC_ABCNT_n[31:0] BCNT and/or EDMA_TPCC_CCNT_n[15:0] CCNT) are this value, the next TR results in:

- Final chaining or interrupt codes sent by the transfer controllers (instead of intermediate).
- Link updates (linking to either null or another valid link set).

Table 16-94. 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	EDMA_TPCC_ABCNT_n[31:0] BCNT == 1 && EDMA_TPCC_CCNT_n[15:0] CCNT == 1
AB-synchronized	ACNT BCNT CCNT	CCNT TRs for ACNT × BCNT bytes each	EDMA_TPCC_CCNT_n[15:0] CCNT == 1

The PaRAM OPT field must program with a specific transfer completion code TCC or EDMA_TPCC_OPT_n[17:12] TCC along with the other EDMA_TPCC_OPT_n fields ([22] TCCHEN, [20] TCINTEN, [23] ITCCHEN, and [21] 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 EDMA_TPCC_OPT_n[17:12] TCC value (6-bit binary value) programmed dictates which of the 64-bits in the chain event register EDMA_TPCC_CER [TCC] and/or interrupt pending register EDMA_TPCC_IPR [TCC] is set.

It can selectively program whether the transfer controller sends back completion codes on completion of the final transfer request (TR) of a parameter set EDMA_TPCC_OPT_n[22] TCCHEN or EDMA_TPCC_OPT_n[20] TCINTEN, for all but the final transfer request (TR) of a parameter set EDMA_TPCC_OPT_n[23] ITCCHEN or EDMA_TPCC_OPT_n[21] ITCINTEN), or for all TRs of a parameter set (both). Refer to Section 16.2.5.8 *Chaining EDMA Channels* for details on chaining (intermediate/final chaining) and Section 16.2.5.9 *EDMA Interrupts* for details on intermediate/final interrupt completion.

A completion detection interface exists between the EDMA 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. Completion of a transfer is used for generating chained events and/or generating interrupts to the CPU(s).

All DMA/QDMA PaRAM sets must also specify a link address value. For repetitive transfers such as ping-pong buffers, the link address value must point to another predefined PaRAM set. Alternatively, a non-repetitive transfer must set the link address value to the null link value. The null link value is defined as FFFFh. Refer to Section 16.2.5.3.7 *Linking Transfers* for more details.

NOTE: Any incoming events that are mapped to a null PaRAM set results in an error condition. The error condition must clear before the corresponding channel is used again. Refer to Section 16.2.5.3.5 *Dummy Versus Null Transfer Comparison*.

There are three ways the EDMA_TPCC 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.

16.2.5.5.1 Normal Completion

In normal completion mode EDMA_TPCC_OPT_n[11] TCCMODE = 0, the transfer or sub-transfer is considered to be complete when the EDMA channel controller receives the completion codes from the EDMA 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.

16.2.5.5.2 Early Completion

In early completion mode EDMA_TPCC_OPT_n[11] TCCMODE = 1, the transfer is considered to be complete when the EDMA channel controller submits the transfer request (TR) to the EDMA 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.

16.2.5.5.3 Dummy or Null Completion

This is a variation of early completion. Dummy or null completion is associated with a dummy set Section 16.2.5.3.4 or null set Section 16.2.5.3.3. In both cases, the EDMA channel controller does not submit the associated transfer request to the EDMA 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 sets the appropriate bits in the interrupt pending registers EDMA_TPCC_IPR and EDMA_TPCC_IPRH or chained event register EDMA_TPCC_CER and EDMA_TPCC_CERH. The internal early completion path is used by the channel controller to return the completion codes internally (that is, EDMA_TPCC generates the completion code).

16.2.5.6 Event, Channel, and PaRAM Mapping

Several of the 64 DMA channels are tied to a specific hardware event, thus allowing events from device peripherals or external hardware (via the dma_evt[4:1] pins) to trigger transfers. 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 (EDMA_TPCC_ABCNT_n[15:0] ACNT, EDMA_TPCC_ABCNT_n[31:16] BCNT, EDMA_TPCC_CCNT_n[15:0] CCNT, etc.) and the synchronization type (A-synchronized or AB-synchronized).

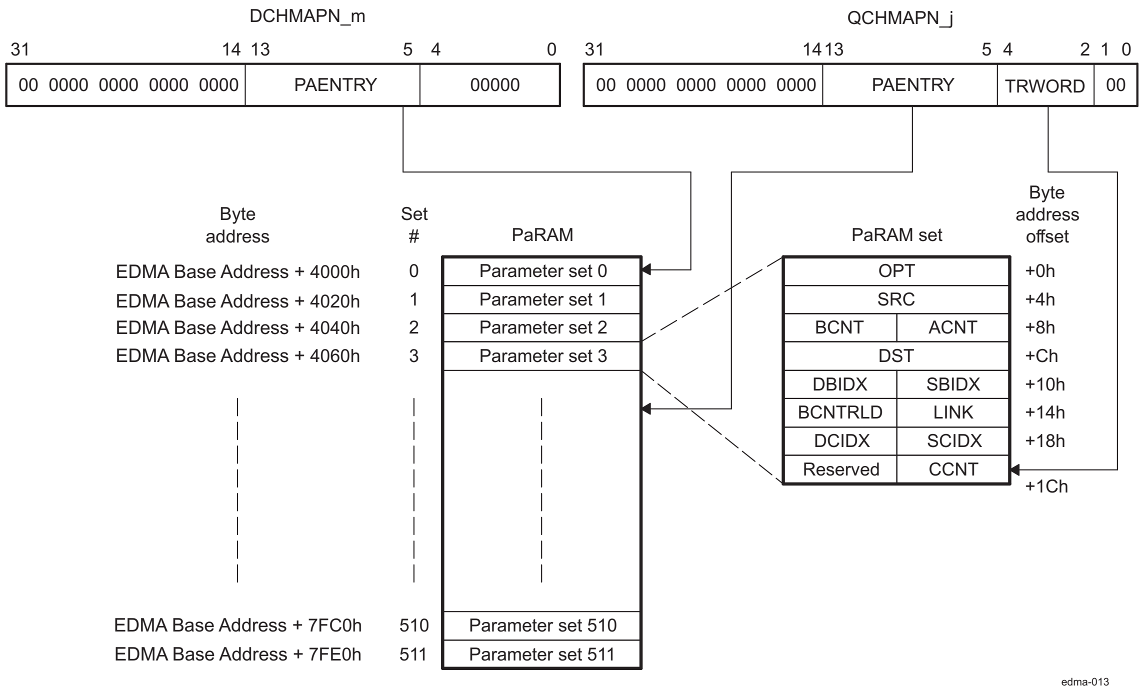
The association of an event to a channel is fixed within the EDMA Channel Controller, that is, each DMA channel has one specific event associated with it. The EDMA event crossbar can be used to select which level events (of which there are more than 64) are mapped to the 64 input events to the EDMA Channel Controller. The default mapping and event crossbar mapping are defined in Section 16.2.4.1, *EDMA Requests to the EDMA Controller*. The event crossbar mapping is controlled by the device Control Modules registers refer to Section 18.4.6.4 *IRQ_CROSSBAR Module Functional Description*, in Chapter 18 *Control Module*.

In an application, if a channel does not use the associated synchronization event or if it 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.

16.2.5.6.1 DMA Channel to PaRAM Mapping

The mapping between the DMA channel numbers and the PaRAM sets is programmable (see Table 16-90). The DMA channel mapping registers EDMA_TPCC_DCHMAPN_m in the EDMA_TPCC provide programmability that allows the DMA channels to be mapped to any of the PaRAM sets in the PaRAM memory map. Figure 16-24 illustrates the use of EDMA_TPCC_DCHMAPN_m. There is one EDMA_TPCC_DCHMAPN_m register per channel.

Figure 16-24. DMA Channel and QDMA Channel to PaRAM Mapping



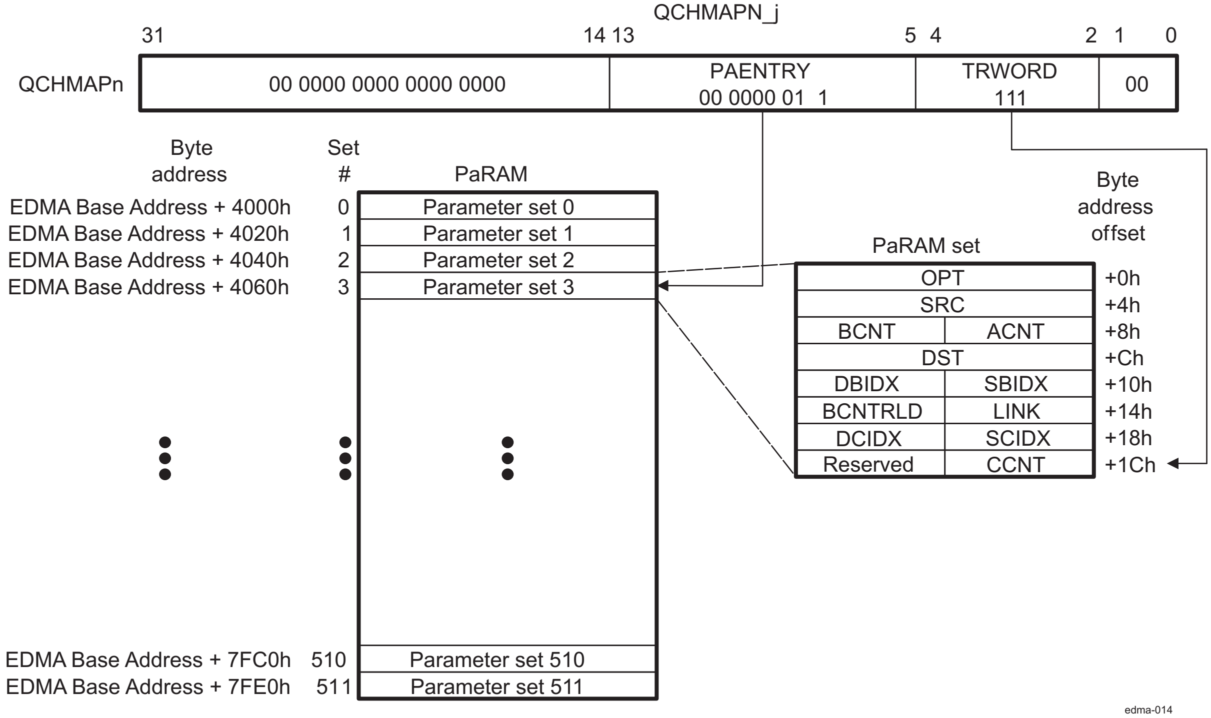
16.2.5.6.2 QDMA Channel to PaRAM Mapping

The mapping between the QDMA channels and the PaRAM sets is programmable. The QDMA channel mapping register EDMA_TPCC_QCHMAPN_j in the EDMA_TPCC allows to map the QDMA channels to any of the PaRAM sets in the PaRAM memory map. Figure 16-25 illustrates the use of EDMA_TPCC_QCHMAPN_j.

EDMA_TPCC_QCHMAPN_j[4:2] TRWORD bit-field allows to program the trigger word in the PaRAM set for the QDMA channel. A trigger word is one of the eight words in the PaRAM set. For a QDMA transfer to occur, a valid TR synchronization event for EDMA_TPCC is a write to the trigger word in the PaRAM set pointed to by EDMA_TPCC_QCHMAPN_j for a particular QDMA channel. By default, QDMA channels are mapped to PaRAM set 0.

It must appropriately re-map PaRAM set 0 before use.

Figure 16-25. QDMA Channel to PaRAM Mapping



16.2.5.7 EDMA Channel Controller Regions

The EDMA channel controller divides its address space into eight regions. Individual channel resources are assigned to a specific region, where each region is typically assigned to a specific device module uses the EDMA controller.

Application software can use regions or to ignore them altogether. It can be used active memory protection in conjunction with regions so that only a specific device module which uses the EDMA (for example, privilege identification) or privilege level (for example, user vs. supervisor) is allowed access to a given region, and thus to a given DMA or QDMA channel. This allows robust system-level DMA code where each EDMA initiator only modifies the state of the assigned resources. Memory protection is described in Section 16.2.5.10 *Memory Protection*.

16.2.5.7.1 Region Overview

The EDMA channel controller 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 EDMA_TPCC memory map. These registers control EDMA resource mapping and provide debug visibility and error tracking information.

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 EDMA_TPCC_EER is visible at the global address of EDMA Base Address + 1020h or region addresses of EDMA Base Address + 2020h for region 0, EDMA Base Address + 2220h for region 1, … EDMA Base Address + 2E20h for region 7.

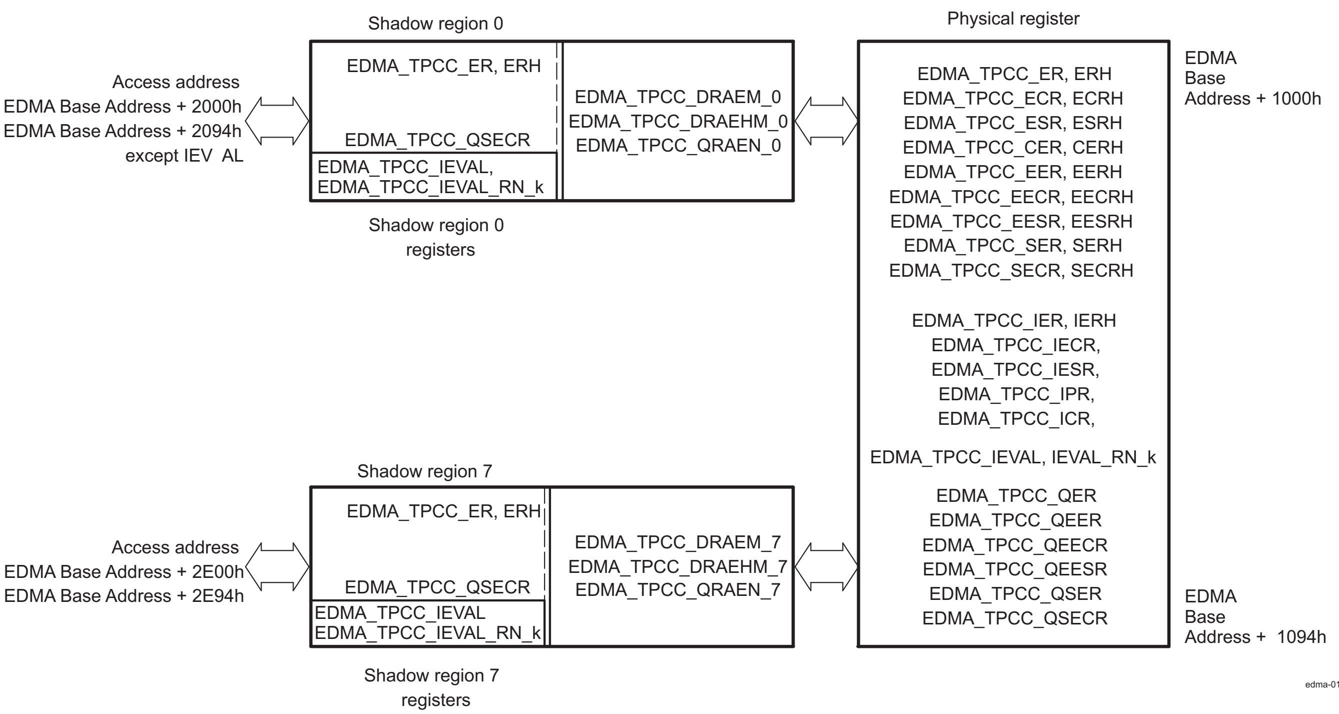
The DMA region access enable registers EDMA_TPCC_DRAEM_k and the QDMA region access enable registers EDMA_TPCC_QRAEN_k control the underlying control register bits that are accessible via the shadow region address space (except for EDMA_TPCC_IEVAL and EDMA_TPCC_IEVAL_RN_k registers). Table 16-95 lists the registers in the shadow region memory map. Refer to EDMA_TPCC register summary Table 16-111 for the complete global and shadow region memory maps.

Table 16-95. Shadow Region Registers

EDMA_TPCC_DRAEM_k	EDMA_TPCC_DRAEHM_k	EDMA_TPCC_QRAEN_k
EDMA_TPCC_ER	EDMA_TPCC_ERH	EDMA_TPCC_QER
EDMA_TPCC_ECR	EDMA_TPCC_ECRH	EDMA_TPCC_QEER
EDMA_TPCC_ESR	EDMA_TPCC_ESRH	EDMA_TPCC_QEECR
EDMA_TPCC_CER	EDMA_TPCC_CERH	EDMA_TPCC_QEESR
EDMA_TPCC_EER	EDMA_TPCC_EERH	
EDMA_TPCC_EECR	EDMA_TPCC_EECRH	
EDMA_TPCC_EESR	EDMA_TPCC_EESRH	
EDMA_TPCC_SER	EDMA_TPCC_SERH	
EDMA_TPCC_SECR	EDMA_TPCC_SECRH	
EDMA_TPCC_IER	EDMA_TPCC_IERH	
EDMA_TPCC_IECR	EDMA_TPCC_IECRH	
EDMA_TPCC_IESR	EDMA_TPCC_IESRH	
EDMA_TPCC_IPR	EDMA_TPCC_IPRH	
EDMA_TPCC_ICR	EDMA_TPCC_ICRH	
Register not affected by DRAE\DRAEH		
EDMA_TPCC_IEVAL		
EDMA_TPCC_IEVAL_RN_k		

Figure 16-26 illustrates the conceptual view of the regions.

Figure 16-26. Shadow Region Registers



16.2.5.7.2 Channel Controller Regions

There are eight EDMA shadow regions (and associated memory maps). Associated with each shadow region are a set of registers defining which channels and interrupt completion codes belong to that region. These registers are user-programmed per region to assign ownership of the DMA/QDMA channels to a region.

- EDMA_TPCC_DRAEM_k and EDMA_TPCC_DRAEHM_k: One register pair exists for each of the shadow regions. The number of bits in each register pair matches the number of DMA channels (64 DMA channels). These registers need to be programmed to assign ownership of DMA channels and interrupt (or EDMA_TPCC_OPT_n[17:12] TCC codes) to the respective region. Accesses to DMA and interrupt registers via the shadow region address view are filtered through the DRAEM/DRAEHM pair. A value of 1 in the corresponding EDMA_TPCC_DRAEM_k[31:0] / EDMA_TPCC_DRAEHM_k[31:0] bit implies that the corresponding DMA interrupt channel is accessible; a value of 0 in the corresponding EDMA_TPCC_DRAEM_k[31:0] / EDMA_TPCC_DRAEHM_k[31:0] bit forces writes to be discarded and returns a value of 0 for reads.
- EDMA_TPCC_QRAEN_k: One register exists for every region. The number of bits in each register matches the number of QDMA channels (4 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 EDMA_TPCC_QEER, the corresponding bits in QRAE must be set or writing into EDMA_TPCC_QEESR there will be no the desired effect.
- EDMA_TPCC_MPPAN_k and EDMA_TPCC_MPPAG: One register exists for every region. This register defines the privilege level, requestor, and types of accesses allowed to a region's memory-mapped registers.

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 restricted access to EDMA resources (DMA channels, QDMA channels, TCC, interrupts) by tasks in a system by setting or clearing bits in the EDMA_TPCC_DRAEM_k / EDMA_TPCC_QRAEN_k registers.

If exclusive access to any given channel / TCC code is required for a region, then only that region's EDMA_TPCC_DRAEM_k / EDMA_TPCC_QRAEN_k have the associated bit set.

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 32 TCC codes (0-15 and 48-63).

Region 1 needs to be allocated 16 DMA channels (16-32) and the remaining 7 QDMA channels (1-7) and TCC codes (16-47).

EDMA_TPCC_DRAEM_k should be equal to the OR of the bits that are required for the DMA channels and the TCC codes:

```
Region 0: DRAEHM, DRAEM = 0xFFFF0000, 0x0000FFFF QRAEN = 0x0000001
Region 1: DRAEHM, DRAEM = 0x0000FFFF, 0xFFFF0000 QRAEN = 0x00000FE
```

16.2.5.7.3 Region Interrupts

In addition to the EDMA_TPCC global completion interrupt, there is an additional completion interrupt line that is associated with every shadow region. Along with the interrupt enable register EDMA_TPCC_IER, DRAEM acts as a secondary interrupt enable for the respective shadow region interrupts. Refer to Table 16-88 *Hardware Request* for more information about EDMA Interrupts.

16.2.5.8 Chaining EDMA Channels

The channel chaining capability for the EDMA allows the completion of an EDMA channel transfer to trigger another EDMA channel transfer. The purpose is to allow the ability to chain several events through one event occurrence.

Chaining is different from linking (Section 16.2.5.3.7 *Linking Transfers*). The EDMA link feature reloads the current channel parameter set with the linked parameter set. The EDMA chaining feature does not modify or update any channel parameter set. It provides a synchronization event to the chained channel (see Section 16.2.5.4.1.3 *Chain-Triggered Transfer Request*).

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* (0-63) needs to be programmed into the EDMA_TPCC_OPT_n[17:12] TCC bit-field of channel *m* channel options parameter (OPT) set.

- If final transfer completion chaining EDMA_TPCC_OPT_n[22] TCCHEN = 1 is enabled, the chain-triggered event occurs after the submission of the last transfer request of channel *m* is either submitted or completed (depending on early or normal completion).
- If intermediate transfer completion chaining EDMA_TPCC_OPT_n[23] ITCCHEN = 1 is enabled, the chain-triggered event occurs after every transfer request, except the last of channel *m* is either submitted or completed (depending on early or normal completion).
- If both final and intermediate transfer completion chaining (EDMA_TPCC_OPT_n[22] TCCHEN = 1 and EDMA_TPCC_OPT_n[23] ITCCHEN = 1) are enabled, then the chain-trigger event occurs after every transfer request is submitted or completed (depending on early or normal completion).

Table 16-96 illustrates the number of chain event triggers occurring in different synchronized scenarios. Consider channel 31 programmed with EDMA_TPCC_ABCNT_n[15:0] ACNT = 3, EDMA_TPCC_ABCNT_n[31:16] BCNT = 4, EDMA_TPCC_CCNT_n[15:0] CCNT = 5, and EDMA_TPCC_OPT_n[17:12] TCC = 30.

Table 16-96. Chain Event Triggers

	(Number of chained event triggers on channel 30)	
Options	**A-Synchronized**	**AB-Synchronized**
EDMA_TPCC_OPT_n[22] TCCHEN = 1, EDMA_TPCC_OPT_n[23] ITCCHEN = 0	1 (Owing to the last TR)	1 (Owing to the last TR)
EDMA_TPCC_OPT_n[22] TCCHEN = 0, EDMA_TPCC_OPT_n[23] ITCCHEN = 1	19 (Owing to all but the last TR)	4 (Owing to all but the last TR)
EDMA_TPCC_OPT_n[22] TCCHEN = 1, EDMA_TPCC_OPT_n[23] ITCCHEN = 1	20 (Owing to a total of 20 TRs)	5 (Owing to a total of 5 TRs)

16.2.5.9 EDMA Interrupts

The EDMA interrupts are divided into 2 categories: transfer completion interrupts and error interrupts.

There are nine region interrupts, eight shadow regions and one global region. The transfer completion interrupts are listed in Table 16-97. The transfer completion interrupts and the error interrupts from the transfer controllers are all routed to the device interrupt controllers INTCs through the inputs of the IRQ_CROSSBAR module.

Table 16-97. EDMA Transfer Completion Interrupts

Name	Description
EDMA_TPCC_INT0	EDMA_TPCC Transfer Completion Interrupt Shadow Region 0
EDMA_TPCC_INT1	EDMA_TPCC Transfer Completion Interrupt Shadow Region 1
EDMA_TPCC_INT2	EDMA_TPCC Transfer Completion Interrupt Shadow Region 2
EDMA_TPCC_INT3	EDMA_TPCC Transfer Completion Interrupt Shadow Region 3
EDMA_TPCC_INT4	EDMA_TPCC Transfer Completion Interrupt Shadow Region 4
EDMA_TPCC_INT5	EDMA_TPCC Transfer Completion Interrupt Shadow Region 5
EDMA_TPCC_INT6	EDMA_TPCC Transfer Completion Interrupt Shadow Region 6
EDMA_TPCC_INT7	EDMA_TPCC Transfer Completion Interrupt Shadow Region 7

Table 16-98. EDMA Error Interrupts

Name	Description
EDMA_TPCC_ERRINT	EDMA_TPCC Error Interrupt
EDMA_TPCC_MPINT	EDMA_TPCC Memory Protection Interrupt
EDMA_TC0_ERRINT	TC0 Error Interrupt
EDMA_TC1_ERRINT	TC1 Error Interrupt

16.2.5.9.1 Transfer Completion Interrupts

The EDMA_TPCC is responsible for generating transfer completion interrupts to the CPU(s) (and other EDMA masters). The EDMA generates a single completion interrupt per shadow region, as well as one for the global region on behalf of all 64 channels. The various control registers and bit fields facilitate EDMA interrupt generation.

The software architecture must either use the global interrupt or the shadow interrupts, but not both.

The transfer completion code EDMA_TPCC_OPT_n[17:12] TCC value is directly mapped to the bits of the interrupt pending register EDMA_TPCC_IPR / EDMA_TPCC_IPRH.

For example, if EDMA_TPCC_OPT_n[17:12] TCC = 10 0001b, EDMA_TPCC_IPRH[1] is set after transfer completion, and results in interrupt generation to the CPU(s) if the completion interrupt is enabled for the CPU. See Section 16.2.5.9.1.1 *Enabling Transfer Completion Interrupts* for details about enabling EDMA transfer completion interrupts.

When a completion code is returned (as a result of early or normal completions), the corresponding bit in EDMA_TPCC_IPR / EDMA_TPCC_IPRH registers is set if transfer completion interrupt (final/intermediate) is enabled in the channel options parameter (OPT) for a PaRAM set associated with the transfer.

Table 16-99. Transfer Complete Code (TCC) to EDMA_TPCC Interrupt Mapping

TCC values in EDMA_TPCC_OPT_n[17:12] TCC (EDMA_TPCC_OPT_n[20] TCINTEN / EDMA_TPCC_OPT_n[21] ITCINTEN = 1)	EDMA_TPCC_IPR Bit Set	TCC values in EDMA_TPCC_OPT_n[17:12] TCC (EDMA_TPCC_OPT_n[20] TCINTEN / EDMA_TPCC_OPT_n[21] ITCINTEN = 1)	EDMA_TPCC_IPRH Bit Set[(1)]
0	EDMA_TPCC_IPR[0]	20h	EDMA_TPCC_IPR[32] / EDMA_TPCC_IPRH[0]
1	EDMA_TPCC_IPR[1]	21h	EDMA_TPCC_IPR[33] / EDMA_TPCC_IPRH[1]
2h	EDMA_TPCC_IPR[2]	22h	EDMA_TPCC_IPR[34] / EDMA_TPCC_IPRH[2]
3h	EDMA_TPCC_IPR[3]	23h	EDMA_TPCC_IPR[35] / EDMA_TPCC_IPRH[3]
4h	EDMA_TPCC_IPR[4]	24h	EDMA_TPCC_IPR[36] / EDMA_TPCC_IPRH[4]
...	...	...	...
1Eh	EDMA_TPCC_IPR[30]	3Eh	EDMA_TPCC_IPR[62] / EDMA_TPCC_IPRH[30]
1Fh	EDMA_TPCC_IPR[31]	3Fh	EDMA_TPCC_IPR[63] / EDMA_TPCC_IPRH[31]

(1) Bit fields EDMA_TPCC_IPR [32-63] correspond to bits 0 to 31 in EDMA_TPCC_IPRH, respectively.

The transfer completion code (TCC) can program to any value for a DMA/QDMA channel. A direct relation between the channel number and the transfer completion code value does not need to exist. 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.

If the channel is used in the context of a shadow region and it intends for the shadow region interrupt to be asserted, then ensure that the bit corresponding to the TCC code is enabled in EDMA_TPCC_IER / EDMA_TPCC_IERH and in the corresponding shadow region's DMA region access registers (EDMA_TPCC_DRAEM_k / EDMA_TPCC_DRAEHM_k).

Interrupt generation can be enabled at either final transfer completion or intermediate transfer completion, or both. Consider channel *m* as an example.

- If the final transfer interrupt (EDMA_TPCC_OPT_n[20] TCINTEN = 1) 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 (EDMA_TPCC_OPT_n[21] ITCINTEN = 1) is enabled, the interrupt occurs after every transfer request, except the last TR of channel *m* is either submitted or completed (depending on early or normal completion).
- If both final and intermediate transfer completion interrupts (EDMA_TPCC_OPT_n[20] TCINTEN = 1, and EDMA_TPCC_OPT_n[21] ITCINTEN = 1) are enabled, then the interrupt occurs after every transfer request is submitted or completed (depending on early or normal completion).

Table 16-100 shows the number of interrupts that occur in different synchronized scenarios. Consider channel 31, programmed with ABCNT_n[15:0] ACNT = 3, EDMA_TPCC_ABCNT_n[31:16] BCNT = 4, EDMA_TPCC_CCNT_n[15:0]CCNT = 5, and EDMA_TPCC_OPT_n[17:12] TCC = 30.

Table 16-100. Number of Interrupts

Options	A-Synchronized	AB-Synchronized
EDMA_TPCC_OPT_n[20] TCINTEN = 1, EDMA_TPCC_OPT_n[21] ITCINTEN = 0	1 (Last TR)	1 (Last TR)
EDMA_TPCC_OPT_n[20] TCINTEN = 0, EDMA_TPCC_OPT_n[21] ITCINTEN = 1	19 (All but the last TR)	4 (All but the last TR)
EDMA_TPCC_OPT_n[20] TCINTEN = 1, EDMA_TPCC_OPT_n[21] ITCINTEN = 1	20 (All TRs)	5 (All TRs)

16.2.5.9.1.1 Enabling Transfer Completion Interrupts

For the EDMA channel controller to assert a transfer completion to the external environment, the interrupts must be enabled in the EDMA_TPCC. This is in addition to setting up the EDMA_TPCC_OPT_n[20] TCINTEN and EDMA_TPCC_OPT_n[21] ITCINTEN bits of the associated PaRAM set.

The EDMA channel controller has interrupt enable registers EDMA_TPCC_IER / EDMA_TPCC_IERH and each bit location in EDMA_TPCC_IER / EDMA_TPCC_IERH serves as a primary enable for the corresponding interrupt pending registers EDMA_TPCC_IPR / EDMA_TPCC_IPRH.

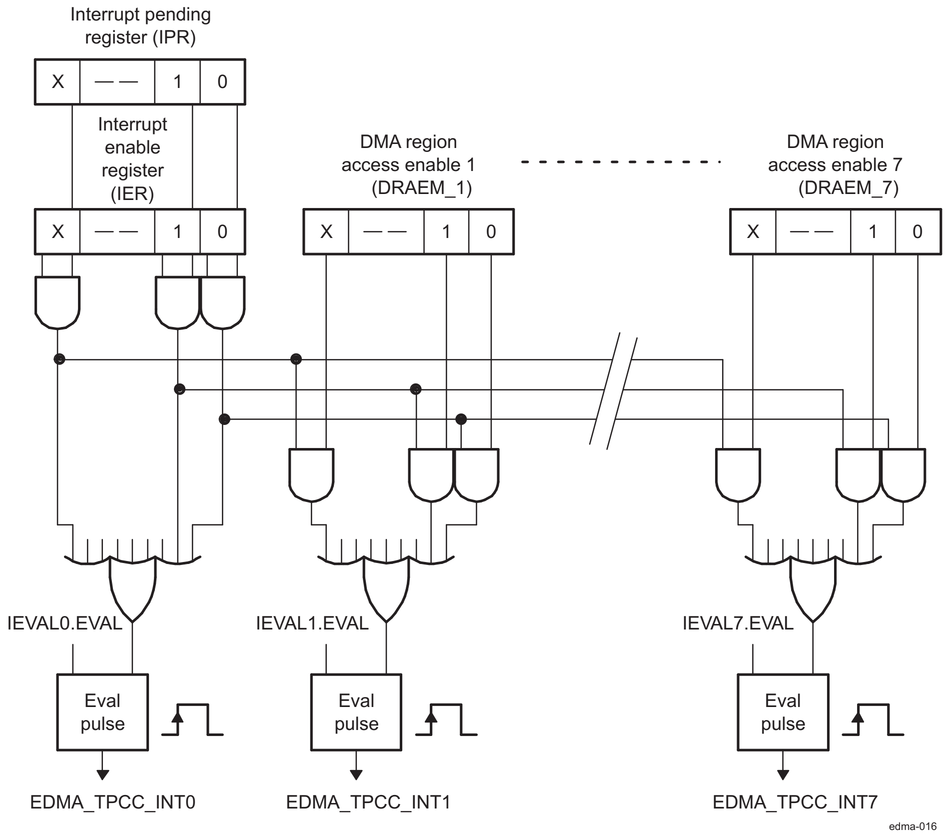
All of the interrupt registers (EDMA_TPCC_IER, EDMA_TPCC_IESR, EDMA_TPCC_IECR, and EDMA_TPCC_IPR) are either manipulated from the global DMA channel region, or by the DMA channel shadow regions. The shadow regions provide a view to the same set of physical registers that are in the global region.

The EDMA channel controller has a hierarchical completion interrupt scheme that uses a single set of interrupt pending registers EDMA_TPCC_IPR / EDMA_TPCC_IPRH and single set of interrupt enable registers EDMA_TPCC_IER / EDMA_TPCC_IERH. The programmable DMA region access enable registers EDMA_TPCC_DRAEM_k / EDMA_TPCC_DRAEHM_k provides a second level of interrupt masking. The global region interrupt output is gated based on the enable mask that is provided by EDMA_TPCC_IER / EDMA_TPCC_IERH, see Figure 16-27

The region interrupt outputs are gated by EDMA_TPCC_IER and the specific EDMA_TPCC_DRAEM_k / EDMA_TPCC_DRAEHM_k associated with the region.

Figure 16-27 shows the Interrupt diagram of the EDMA controller.

Figure 16-27. Interrupt Diagram



The EDMA_TPCC generates the transfer completion interrupts that are associated with each shadow region, the following conditions must be true:

- EDMA_TPCC_INT0: (EDMA_TPCC_IPR[0] E0 & EDMA_TPCC_IER[0] E0 & EDMA_TPCC_DRAEM_k.DRAEM_0[0] E0) | (EDMA_TPCC_IPR[1] E1 & EDMA_TPCC_IER[1] E1 & EDMA_TPCC_DRAEM_k.DRAEM_0[1] E1) | …|(EDMA_TPCC_IPRH[31] E63 & EDMA_TPCC_IERH[31] E63 & EDMA_TPCC_DRAEHM_k.DRAEHM_0[31] E63)
- EDMA_TPCC_INT1: (EDMA_TPCC_IPR[0] E0 & EDMA_TPCC_IER[0] E0 & EDMA_TPCC_DRAEM_k.DRAEM_1[0] E0) | (EDMA_TPCC_IPR[1] E1 & EDMA_TPCC_IER[1] E1 & EDMA_TPCC_DRAEM_k.DRAEM_1[1] E1) | …| (EDMA_TPCC_IPRH[31] E63 & EDMA_TPCC_IERH[31] E63 & EDMA_TPCC_DRAEHM_k.DRAEHM_1[31] E63)
- EDMA_TPCC_INT2: (EDMA_TPCC_IPR[0] E0 & EDMA_TPCC_IER[0] E0 & EDMA_TPCC_DRAEM_k.DRAEM_2[0] E0) | (EDMA_TPCC_IPR[1] E1 & EDMA_TPCC_IER[1] E1 & EDMA_TPCC_DRAEM_k.DRAEM_2[1] E1) | …|(EDMA_TPCC_IPRH[31] E63 & EDMA_TPCC_IERH[31] E63 & EDMA_TPCC_DRAEHM_k.DRAEHM_2[31] E63)....
- Up to EDMA_TPCC_INT7: (EDMA_TPCC_IPR[0] E0 & EDMA_TPCC_IER[0] E0 & EDMA_TPCC_DRAEM_k.DRAEM_7[0] E0) | (EDMA_TPCC_IPR[1] E1 & EDMA_TPCC_IER[1] E1 & EDMA_TPCC_DRAEM_k.DRAEM_7[1] E1) | …|(EDMA_TPCC_IPRH[31] E63 & EDMA_TPCC_IERH[31] E63 & EDMA_TPCC_DRAEHM_k.DRAEHM_7[31] E63)

NOTE: The EDMA_TPCC_DRAEM_k / EDMA_TPCC_DRAEHM_k for all regions are expected to be set up at system initialization and to remain static for an extended period of time. The interrupt enable registers are used for dynamic enable/disable of individual interrupts.

Because there is no relation between the EDMA_TPCC_OPT_n[17:12] TCC value and the DMA/QDMA channel, it is possible, the DMA channel 0 to have the EDMA_TPCC_OPT_n[17:12] TCC = 63 in its associated PaRAM set. This mean that if a transfer completion interrupt is enabled (EDMA_TPCC_OPT_n[20] TCINTEN or EDMA_TPCC_OPT_n[21] ITCINTEN is set), then based on the TCC value, EDMA_TPCC_IPRH[31] E63 is set up on completion. For proper channel operations and interrupt generation using the shadow region map - program the EDMA_TPCC_DRAEM_k / EDMA_TPCC_DRAEHM_k that is associated with the shadow region to have read/write access to both bit 0 (corresponding to channel 0) and bit 63 (corresponding to EDMA_TPCC_IPRH bit that is set upon completion).

16.2.5.9.1.2 Clearing Transfer Completion Interrupts

Transfer completion interrupts that are latched to the interrupt pending registers (EDMA_TPCC_IPR / EDMA_TPCC_IPRH) are cleared by writing a 1 to the corresponding bit in the interrupt pending clear register (EDMA_TPCC_ICR / EDMA_TPCC_ICRH). For example, a write of 1 to EDMA_TPCC_ICR[0] E0 clears a pending interrupt in EDMA_TPCC_IPR[0] E0.

If an incoming transfer completion code TCC (EDMA_TPCC_OPT_n[17:12] TCC) gets latched to a bit in EDMA_TPCC_IPR / EDMA_TPCC_IPRH, then additional bits that get set due to a subsequent transfer completion does not result in asserting the EDMA_TPCC 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.

16.2.5.9.2 EDMA Interrupt Servicing

Upon completion of a transfer (early or normal completion), the EDMA channel controller sets the appropriate bit in the interrupt pending registers (EDMA_TPCC_IPR / EDMA_TPCC_IPRH), as the transfer completion codes specify. If the completion interrupts are appropriately enabled, then the CPU enters the interrupt service routine (ISR) when the completion interrupt is asserted.

After servicing the interrupt, the ISR should clear the corresponding bit in EDMA_TPCC_IPR/EDMA_TPCC_IPRH, thereby enabling recognition of future interrupts. The EDMA_TPCC only asserts additional completion interrupts when all EDMA_TPCC_IPR / EDMA_TPCC_IPRH bits clear.

When one interrupt is serviced many other transfer completions may result in additional bits being set in EDMA_TPCC_IPR / EDMA_TPCC_IPRH, thereby resulting in additional interrupts. Each of the bits in EDMA_TPCC_IPR / EDMA_TPCC_IPRH may need different types of service therefore, the ISR must check all pending interrupts and continue until all of the posted interrupts are serviced appropriately.

Examples of pseudo code for a CPU interrupt service routine for an EDMA_TPCC completion interrupt are shown in Example 16-2 and Example 16-3.

The ISR routine in Example 16-2 is more exhaustive and incurs a higher latency.

Example 16-2. Interrupt Servicing

The pseudo code:

1. Reads the interrupt pending register EDMA_TPCC_IPR / EDMA_TPCC_IPRH.
2. Performs the operations needed.
3. Writes to the interrupt pending clear register EDMA_TPCC_ICR / EDMA_TPCC_ICRH to clear the corresponding EDMA_TPCC_IPR / EDMA_TPCC_IPRH bit(s).
4. Reads EDMA_TPCC_IPR / EDMA_TPCC_IPRH again:
   (a) If EDMA_TPCC_IPR / EDMA_TPCC_IPRH is not equal to 0, repeat from step 2 (implies occurrence of new event between step 2 to step 4).

Example 16-2. Interrupt Servicing (continued)

(b) If EDMA_TPCC_IPR / EDMA_TPCC_IPRH is equal to 0, assure that all of the enabled interrupts are inactive.

NOTE: An event may occur during step 4 while the EDMA_TPCC_IPR / EDMA_TPCC_IPRH bits are read as 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 generates as soon as the application exits in the interrupt service routine.

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 any enabled and pending (possibly lower priority) interrupts are left, force the interrupt logic to reassert the interrupt pulse by setting the EDMA_TPCC_IEVAL[0] EVAL bit in the interrupt evaluation register.

The pseudo code is as follows:

1. Enters ISR.
2. Reads EDMA_TPCC_IPR / EDMA_TPCC_IPRH.
3. For the condition that is set in EDMA_TPCC_IPR / EDMA_TPCC_IPRH:
   (a) Service interrupt as the application requires.
   (b) Clear the bit for serviced conditions (others may still be set, and other transfers may have resulted in returning the TCC to EDMA_TPCC after step 2).
4. Reads EDMA_TPCC_IPR / EDMA_TPCC_IPRH prior to exiting the ISR:
   (a) If EDMA_TPCC_IPR / EDMA_TPCC_IPRH is equal to 0, then exit the ISR.
   (b) If EDMA_TPCC_IPR / EDMA_TPCC_IPRH is not equal to 0, then set EDMA_TPCC_IEVAL so that upon exit of ISR, a new interrupt triggers if any enabled interrupts are still pending.

16.2.5.9.3 Interrupt Evaluation Operations

The EDMA_TPCC has interrupt evaluate registers EDMA_TPCC_IEVAL that exist in the global region and in each shadow region. The registers in the shadow region 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 EDMA_TPCC_DRAEM_k / EDMA_TPCC_DRAEHM_k. Writing a 1 to the EDMA_TPCC_IEVAL[0] EVAL bit in the registers that are associated with a particular shadow region results in pulsing the associated region interrupt (global or shadow), if any enabled interrupt (via EDMA_TPCC_IER / EDMA_TPCC_IERH) is still pending EDMA_TPCC_IPR / EDMA_TPCC_IPRH. This register assures that the CPU does not miss the interrupts (or the EDMA master associated with the shadow region) if the software architecture chooses not to use all interrupts. Refer to Example 16-3 about the use of EDMA_TPCC_IEVAL in the EDMA interrupt service routine (ISR).

Similarly an error evaluation register EDMA_TPCC_EEVAL exists in the global region. Writing a 1 to the EDMA_TPCC_EEVAL[0] EVAL bit causes the pulsing of the error interrupt if any pending errors are in EDMA_TPCC_EMR / EDMA_TPCC_EMRH, EDMA_TPCC_QEMR, or EDMA_TPCC_CCERR. See Section 16.2.5.9.4 *Error Interrupts* for additional information regarding error interrupts.

NOTE: While using EDMA_TPCC_IEVAL for shadow region completion interrupts, check that the EDMA_TPCC_IEVAL operated upon is from that particular shadow region memory map.

16.2.5.9.4 Error Interrupts

The EDMA_TPCC error registers provide the capability to differentiate error conditions (event missed, threshold exceed, etc.). Additionally, setting the error bits in these registers results in asserting the EDMA_TPCC error interrupt. If the EDMA_TPCC error interrupt is enabled in the device interrupt controller(s), then it allows the CPU(s) to handle the error conditions.

The EDMA_TPCC has a single error interrupt (EDMA_TPCC_ERRINT) that is asserted for all EDMA_TPCC error conditions. There are four conditions that cause the error interrupt:

- DMA missed events: for all 64 DMA channels. DMA missed events are latched in the event missed registers EDMA_TPCC_EMR / EDMA_TPCC_EMRH.
- QDMA missed events: for all 8 QDMA channels. QDMA missed events are latched in the QDMA event missed register EDMA_TPCC_QEMR.
- Threshold exceed: for all event queues. These are latched in EDMA_TPCC error register EDMA_TPCC_CCERR.
- TCC error: for outstanding transfer requests that are expected to return completion code EDMA_TPCC_OPT_n[22] TCCHEN or EDMA_TPCC_OPT_n[23] TCINTEN bit is set to 1, exceeding the maximum limit of 63. This is also latched in the EDMA_TPCC error register EDMA_TPCC_CCERR.

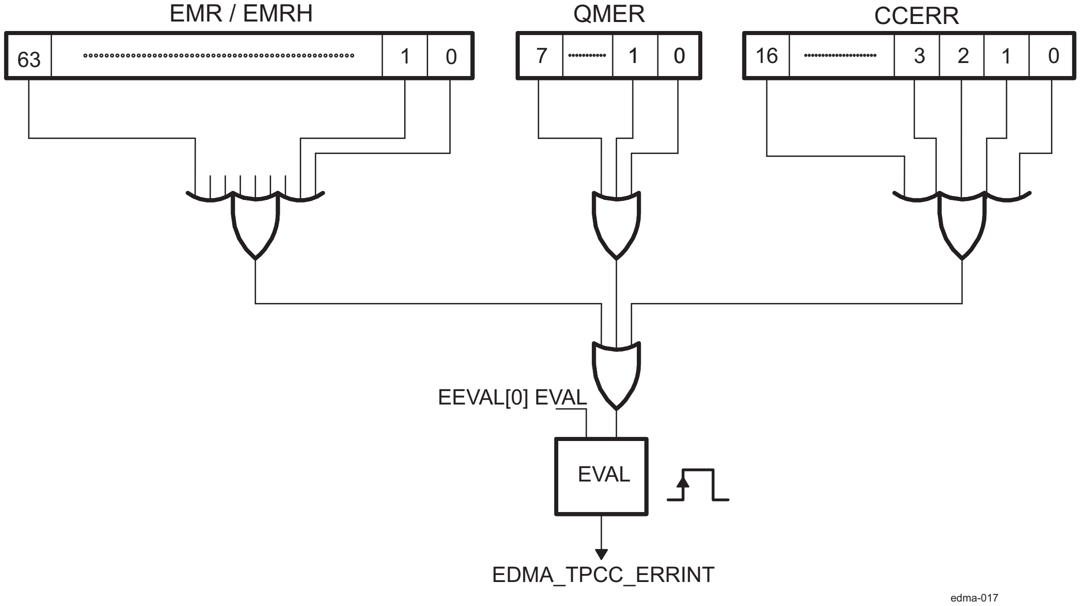
Figure 16-28 illustrates the EDMA_TPCC error interrupt generation operation.

If any of the bits are set in the error registers due to any error condition, the EDMA_TPCC_ERRINT is always asserted, as there are no enables for masking these error events. Similar to transfer completion interrupts (EDMA_TPCC_INT), the error interrupt also only pulses when the error interrupt condition transitions from no errors being set to at least one error being set. If additional error events are latched prior to the original error bits clearing, the EDMA_TPCC does not generate additional interrupt.

To reduce the burden on the software, there is an error evaluate register EDMA_TPCC_EEVAL that allows re-evaluation of pending set error events/bits, similar to the interrupt evaluate register EDMA_TPCC_IEVAL. Unlike the EDMA_TPCC_IEVAL functionality, the EDMA_TPCC_EEVAL register must be written with ‘1’ after any error interrupts are serviced (even when all pending errors are cleared) in order for subsequent errors to trigger a new interrupt.

NOTE: It is good practice to enable the error interrupt in the device interrupt controller and to associate an interrupt service routine with it to address the various error conditions appropriately. Doing so puts less burden on the software (polling for error status), it provides a good debug mechanism for unexpected error conditions.

Figure 16-28. Error Interrupt Operation



16.2.5.10 Memory Protection

The EDMA channel controller supports two kinds of memory protection: active and proxy.

16.2.5.10.1 Active Memory Protection

Active memory protection is a feature that allows or prevents read and write accesses to the EDMA_TPCC registers. Active memory protection is achieved by a set of memory protection permissions attribute EDMA_TPCC_MPPAN_k registers.

The EDMA_TPCC register map is divided into three categories:

- a global region.
- a global channel region.
- eight shadow regions.

Each shadow region consists of the respective shadow region registers and the associated PaRAM. For more detailed information regarding the contents of a shadow region, refer to Table 16-111 *EDMA_TPCC Registers Mapping Summary*.

Each of the eight shadow regions has an associated EDMA_TPCC_MPPAN_k registers that defines the specific requestor(s) and types of requests that are allowed to the regions resources.

The global channel region is also protected with a memory-mapped register EDMA_TPCC_MPPAG. The EDMA_TPCC_MPPAG applies to the global region and to the global channel region, except the other EDMA_TPCC_MPPAN_k registers themselves.

Table 16-101 shows the accesses that are allowed or not allowed to the EDMA_TPCC_MPPAG and EDMA_TPCC_MPPAN_k. The active memory protection uses the EDMA_TPCC_OPT_n[31] PRIV and EDMA_TPCC_OPT_n[27:24] PRIVID attributes of the EDMA peripheral modules. TheEDMA_TPCC_OPT_n[31] PRIV is the privilege level (i.e., user vs. supervisor).

The EDMA_TPCC_OPT_n[27:24] PRIVID refers to a privilege ID with a number that is associated with an EDMA peripheral modules.

Table 16-101. Allowed Accesses

Access	Supervisor	User
Read	Yes	Yes
Write	Yes	No

Table 16-102 describes the EDMA_TPCC_MPPAN_k register mapping for the shadow regions (which includes shadow region registers and PaRAM addresses).

The region-based EDMA_TPCC_MPPAN_k registers are used to protect accesses to the DMA shadow regions and the associated region PaRAM. Because there are eight regions, there are eight EDMA_TPCC_MPPAN_k region registers (MPPA[0-7]).

Table 16-102. MPPA Registers to Region Assignment

Register	Registers Protect	Address Range	PaRAM Protect[(1)]	Address Range
EDMA_TPCC_MPPAG	Global Range	0000h-1FFCh	N/A	N/A
EDMA_TPCC_MPPAN_k. MPPAN_0	DMA Shadow 0	2000h-21FCh	1st octant	4000h-47FCh
MPPAN_1	DMA Shadow 1	2200h-23FCh	2nd octant	4800h-4FFCh
MPPAN_2	DMA Shadow 2	2400h-25FCh	3rd octant	5000h-57FCh
MPPAN_3	DMA Shadow 3	2600h-27FCh	4th octant	5800h-5FFCh
MPPAN_4	DMA Shadow 4	2800h-29FCh	5th octant	6000h-67FCh
MPPAN_5	DMA Shadow 5	2A00h-2BFCh	6th octant	6800h-6FFCh
MPPAN_6	DMA Shadow 6	2C00h-2DFCh	7th octant	7000h-77FCh
MPPAN_7	DMA Shadow 7	2E00h-2FFCh	8th octant	7800h-7FFCh

(1) The PARAM region is divided into 8 regions referred to as an octant.

Example *Access denied.*

Write access to shadow region 7's event enable set register EDMA_TPCC_EESR:

1. The original value of the event enable register EDMA_TPCC_EER at address offset 0x1020 is 0x0.
2. The EDMA_TPCC_MPPAN_k. EDMA_TPCC_MPPAN_7[7] NS is set to prevent user level accesses (EDMA_TPCC_MPPAN_k. EDMA_TPCC_MPPAN_7[1] UW = 0, EDMA_TPCC_MPPAN_k. EDMA_TPCC_MPPAN_7[2] UR = 0), but it allows supervisor level accesses (EDMA_TPCC_MPPAN_k. EDMA_TPCC_MPPAN_7[4] SW = 1, EDMA_TPCC_MPPAN_k. EDMA_TPCC_MPPAN_7[5] SR = 1) with a privilege ID of 0. (EDMA_TPCC_MPPAN_k. EDMA_TPCC_MPPAN_7[10] AID0 = 1).
3. EDMA peripheral modules with a privilege ID of 0 attempts to perform a user-level write of a value of 0xFF00FF00 to shadow region 7's event enable set register EDMA_TPCC_EESR at address offset 0x2E30.

NOTE: The EDMA_TPCC_EER is a read-only register and the only way that write to it is by writing to the EDMA_TPCC_EESR. There is only one physical register for EDMA_TPCC_EER, EDMA_TPCC_EESR, etc. and that the shadow regions only provide to the same physical set.

4. Since the EDMA_TPCC_MPPAN_k. EDMA_TPCC_MPPAN_7[1] UW = 0, though the privilege ID of the write access is set to 0, the access is not allowed and the EDMA_TPCC_EER is not written too.

Table 16-103. Example Access Denied

Register	Value	Description
EDMA_TPCC_EER (offset 0x1020)	0x0000 0000	Value in EDMA_TPCC_EER to begin with.
EDMA_TPCC_EESR (offset 0x2E30)	0xFF00 FF00 ↓	Value attempted to be written to shadow region 7's EDMA_TPCC_EESR. This is done by an EDMA connected device module with a privilege level of User and Privilege ID of 0.
EDMA_TPCC_MPPAN_k (offset 0x082C)	0x0000 04B0	Memory Protection Filter EDMA_TPCC_MPPAN_k[10] AID0 = 1, EDMA_TPCC_MPPAN_k[1] UW = 0, EDMA_TPCC_MPPAN_k[2] UR = 0, EDMA_TPCC_MPPAN_k[4] SW = 1, EDMA_TPCC_MPPAN_k[5] SR = 1.
	X	Access Denied
EDMA_TPCC_EER (offset 0x1020)	0x0000 0000	Final value of EDMA_TPCC_EER

Example *Access Allowed*

Write access to shadow region 7's event enable set register EDMA_TPCC_EESR:

1. The original value of the event enable register EDMA_TPCC_EER at address offset 0x1020 is 0x0.
2. The EDMA_TPCC_MPPAN_k.EDMA_TPCC_MPPAN_7 is set to allow user-level accesses (EDMA_TPCC_MPPAN_k.EDMA_TPCC_MPPAN_7[1] UW = 1, EDMA_TPCC_MPPAN_k. EDMA_TPCC_MPPAN_7[2] UR = 1) and supervisor-level accesses (EDMA_TPCC_MPPAN_k. EDMA_TPCC_MPPAN_7[4] SW = 1, EDMA_TPCC_MPPAN_k. EDMA_TPCC_MPPAN_7[5] SR = 1) with a privilege ID of 0. (EDMA_TPCC_MPPAN_k. EDMA_TPCC_MPPAN_7[10] AID0 = 1).
3. EDMA peripheral modules with a privilege ID of 0, attempts to perform a user-level write of a value of 0xABCD0123 to shadow region 7's event enable set register EDMA_TPCC_EESR at address offset 0x2E30.

NOTE: The EDMA_TPCC_EER is a read-only register and the only way that write to it is by writing to the EDMA_TPCC_EESR. There is only one physical register for EDMA_TPCC_EER, EDMA_TPCC_EESR, etc. and that the shadow regions only provide to the same physical set.

4. Since the EDMA_TPCC_MPPAN_k. EDMA_TPCC_MPPAN_7[1] UW = 1 and EDMA_TPCC_MPPAN_k. MPPAN_7[10] AID0 = 1, the user-level write access is allowed.
5. The accesse to shadow region registers are masked by their respective EDMA_TPCC_DRAEM_k

register. In this example, the EDMA_TPCC_DRAEM_k. EDMA_TPCC_DRAEM_7 is set of 0x9FF00FC2.

6. The value finally written to EDMA_TPCC_EER is 0x8BC00102.

Table 16-104. Example Access Allowed

Register	Value	Description
EDMA_TPCC_EER (offset 0x1020)	0x0000 0000	Value in EER to begin with.
EDMA_TPCC_EESR (offset 0x2E30)	0xFF00 FF00	Value attempted to be written to shadow region 7's EESR. This is done by an EDMA peripheral module with a privilege level of User and Privilege ID of 0.
EDMA_TPCC_MPPAN_k. EDMA_TPCC_MPPAN_7 (offset 0x082C)	0x0000 04B3	Memory Protection Filter EDMA_TPCC_MPPAN_k[10] AID = 1, EDMA_TPCC_MPPAN_k. EDMA_TPCC_MPPAN_7[1] UW = 1, EDMA_TPCC_MPPAN_k. EDMA_TPCC_MPPAN_7[2] UR = 1, EDMA_TPCC_MPPAN_k. EDMA_TPCC_MPPAN_7[4] SW = 1, EDMA_TPCC_MPPAN_k. EDMA_TPCC_MPPAN_7[5] SR = 1.
	√ ↓	Access allowed.
EDMA_TPCC_DRAEM_k. EDMA_TPCC_DRAEM_7 (offset 0x0378)	0x9FF0 0FC2 ↓	DMA Region Access Enable Filter
EDMA_TPCC_EESR (offset 0x2E30)	0x8BC0 0102 ↓	Value written to shadow region 7's EESR. This is done by an EDMA peripheral module with a privilege level of User and a Privilege ID of 0.
EDMA_TPCC_EER (offset 0x1020)	↓ 0xBC0 0102	Final value of EER.

16.2.5.10.2 Proxy Memory Protection

Proxy memory protection allows an EDMA transfer programmed by a given peripheral module connected to EDMA, to have its permissions travel with the transfer through the EDMA_TPTC. The permissions travel along with the read transactions to the source and the write transactions to the destination endpoints. The EDMA_TPCC_OPT_n[31] PRIV bit and EDMA_TPCC_OPT_n[27:24] PRIVID bit is set with the peripheral module's PRIV value and PRIVID values, respectively, when any part of the PaRAM set is written.

The EDMA_TPCC_OPT_n[31] PRIV is the privilege level (i.e., user vs. supervisor). The EDMA_TPCC_OPT_n[27:24] PRIVID refers to a privilege ID with a number that is associated with an peripheral module connected to EDMA.

These options are part of the TR that are submitted to the transfer controller. The transfer controller uses the above values on their respective read and write command bus so that the target endpoints can perform memory protection checks based on these values.

Consider a parameter set that is programmed by a CPU in user privilege level for a simple transfer with the source buffer on an L2 page and the destination buffer on an L1D page. The EDMA_TPCC_OPT_n[31] PRIV is 0 for user-level and the CPU has a EDMA_TPCC_OPT_n[27:24] PRIVID to 0.

The PaRAM set is shown in Figure 16-29.

Figure 16-29. PaRAM Set Content for Proxy Memory Protection Example

(a) EDMA Parameters

Parameter Contents		Parameter	
0010 0007h		Channel Options Parameter (OPT)	
009F 0000h		Channel Source Address (SRC)	
0001h	0004h	Count for 2nd Dimension (BCNT)	Count for 1st Dimension (ACNT)
00F0 7800h		Channel Destination Address (DST)	
0001h	0001h	Destination BCNT Index (DBIDX)	Source BCNT Index (SBIDX)
0000h	FFFFh	BCNT Reload (BCNTRLD)	Link Address (LINK)
0001h	1000h	Destination CCNT Index (DCIDX)	Source CCNT Index (SCIDX)
0000h	0001h	Reserved	Count for 3rd Dimension (CCNT)

Figure 16-30. Channel Options Parameter (OPT) Example

(b) Channel Options Parameter (OPT_n) Content

31	30	29–28	27–24	23	22	21	20	19–18	17–16
0	0	00	0000	0	0	0	1	00	00
PRIV	Rsvd	Rsvd	PRIVID	ITCCHEN	TCCHEN	ITCINTEN	TCINTEN	Reserved	TCC

15–12	11	10–8	7–4	3	2	1	0
0000	0	000	0000	0	1	1	1
TCC	TCCMOD	FWID	Reserved	STATIC	SYNCDIM	DAM	SAM

edma-018

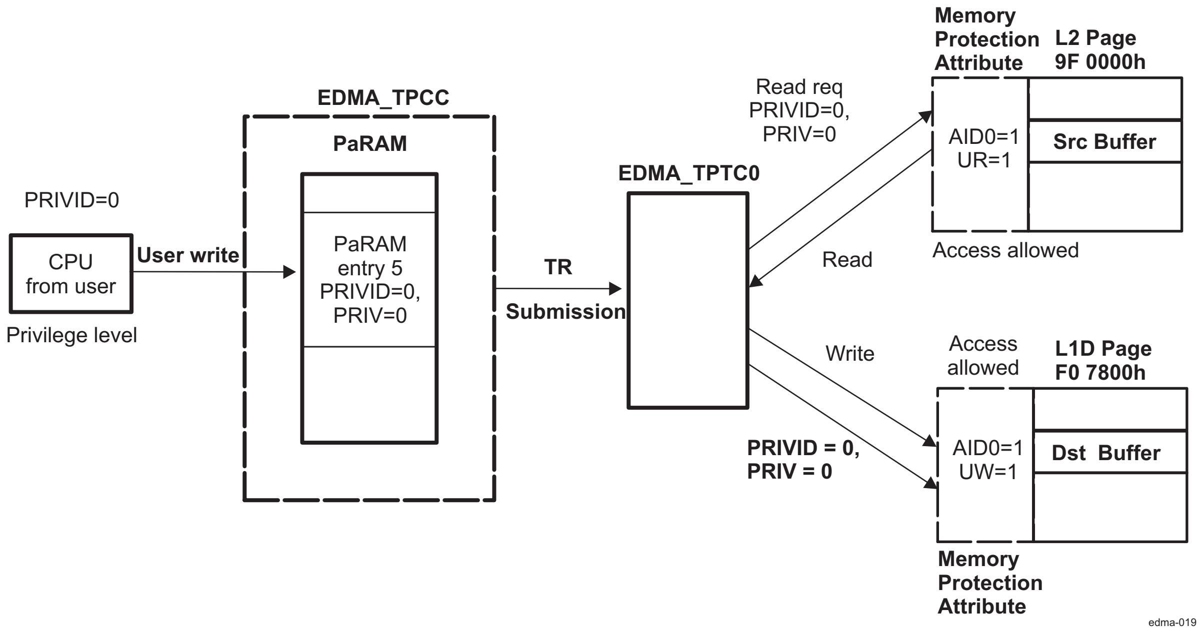
The EDMA_TPCC_OPT_n[31] PRIV and EDMA_TPCC_OPT_n[27:24] PRIVID information travels along with the read and write requests that are issued to the source and destination memories.

For example, if the access attributes that are associated with the L2 page with the source buffer only allow supervisor read, write accesses EDMA_TPCC_MPPAN_k[4] SW and EDMA_TPCC_MPPAN_k[5] SR, the user-level read request above is refused. Similarly, if the access attributes that are associated with the L1D page with the destination buffer only allow supervisor read and write accesses (EDMA_TPCC_MPPAN_k[4] SW, EDMA_TPCC_MPPAN_k[5] SR), the user-level write request above is refused. For the transfer to succeed, the source and destination pages must have user-read and user-write permissions, respectively, along with allowing accesses from a PRIVID = 0.

Because the privilege level and privilege identification travel with the read and write requests, EDMA acts as a proxy.

Figure 16-31 illustrates the propagation of EDMA_TPCC_OPT_n[31] PRIV and EDMA_TPCC_OPT_n[27:24] PRIVID at the boundaries of all the interacting entities (CPU, EDMA_TPCC, EDMA_TPTCs, and slave memories).

Figure 16-31. Proxy Memory Protection Example



16.2.5.11 Event Queue(s)

Event queues are a part of the EDMA channel controller. Event queues form the interface between the event detection logic in the EDMA_TPCC and the transfer request (TR) submission logic of the EDMA_TPCC. Each queue is 16 entries deep. Each event queue can queue a maximum of 16 events. 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 and the CPU does not stall.

There are two event queues for the device: Queue0, Queue1. Events in Queue0 result in submission of its associated transfer requests (TRs) to TC0. The transfer requests that are associated with events in Queue1 are submitted to TC1.

An event that wins prioritization against other DMA and/or QDMA pending events is placed at the tail of the appropriate event queue. Each event queue is serviced in FIFO order. Once the event reaches the head of its queue and the corresponding transfer controller is ready to receive another TR, the event is de-queued and the PaRAM set corresponding to the de-queued event is processed and submitted as a transfer request packet (TRP) to the associated EDMA transfer controller.

Queue0 has highest priority and Queue1 has the lowest priority, if Queue0 and Queue1 both have at least one event entry and if both TC0 and TC1 can accept transfer requests, then the event in Queue0 is de-queued first and its associated PaRAM set is processed and submitted as a transfer request (TR) to TC0.

Refer to Section 16.2.5.11.4 for system-level performance considerations. All of the event entries in all of the event queues are software readable (not writeable) by accessing the event entry registers EDMA_TPCC_Q0E_p and EDMA_TPCC_Q1E_p. Each event entry register characterizes the queued event in terms of the type of event (manual, event, chained or auto-triggered) and the event number. Refer to Section 16.2.8.2.2.1 *EDMA_TPCC Register Description* for Q0E_p / Q1E_p descriptions of the bit fields.

16.2.5.11.1 DMA/QDMA Channel to Event Queue Mapping

Each of the 64 DMA channels and eight QDMA channels are programmed independently to map to a specific queue, using the DMA queue number register EDMA_TPCC_DMAQNUMN_k and the QDMA queue number register EDMA_TPCC_QDMAQNUM. 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. Refer to Section 16.2.5.11.4 *System-level Performance Considerations*.

NOTE: If an event is ready to be queued and both the event queue and the EDMA transfer controller that is associated to the event queue are empty, then the event bypasses the event queue, and moves the PaRAM processing logic, and eventually to the transfer request submission logic for submission to the EDMA_TPTC. In this case, the event is not logged in the event queue status registers.

16.2.5.11.2 Queue RAM Debug Visibility

There are two event queues and each queue has 16 entries. These 16 entries are managed in a circular FIFO. There is a queue status register EDMA_TPCC_QSTATN_i associated with each queue. These along with all of the 16 entries per queue can be read via registers EDMA_TPCC_QSTATN_i and Q0E_p / Q1E_p, respectively.

These registers provide user visibility.

The event queue entry register (Q*x*E*y* Q0E_p / Q1E_p) uniquely identifies the specific event type (event-triggered, manually-triggered, chain-triggered, and QDMA events) along with the event number (for all DMA/QDMA event channels) that are in the queue or have been de-queued (passed through the queue).

Each of the 16 entries in the event queue are read using the EDMA_TPCC memory-mapped register. To see the history of the last 16 TRs that have been processed by the EDMA on a given queue, read the event queue registers. This provides user/software visibility and is helpful for debugging real-time issues (typically post-mortem), involving multiple events and event sources.

The queue status register (QSTAT*n* EDMA_TPCC_QSTATN_i) includes fields for the start pointer EDMA_TPCC_QSTATN_i[3:0] STRTPTR which provides the offset to the head entry of an event. It also includes a field called EDMA_TPCC_QSTATN_i[12:8] NUMVAL that provides the total number of valid entries residing in the event queue at a given instance of time. The EDMA_TPCC_QSTATN_i[3:0] STRTPTR is used to index appropriately into the 16 event entries. EDMA_TPCC_QSTATN_i[12:8] NUMVAL number of entries starting from STRTPTR are indicative of events still queued in the respective queue. The remaining entry must be read to determine what's already de-queued and submitted to the associated transfer controller.

16.2.5.11.3 Queue Resource Tracking

The EDMA_TPCC event queue includes watermarking/threshold logic that allows 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 EDMA event queue.

The maximum number of events are programed that the queue up in an event queue by programming the threshold value (between 0 to 15) in the queue watermark threshold A register EDMA_TPCC_QWMTHRA. The maximum queue usage is recorded actively in the watermark EDMA_TPCC_QSTATN_i[20:16] WM field of the queue status register, that keeps getting updated based on a comparison of number of valid entries, which is also visible in the EDMA_TPCC_QSTATN_i[12:8] NUMVAL bit and the maximum number of entries.

If the queue usage is exceeded, this status is visible in the EDMA_TPCC registers: the QTHRXCD*n* bits in the channel controller error register EDMA_TPCC_CCERR[7:0] and the EDMA_TPCC_QSTATN_i[24] THRXCD bit, where *n* stands for the event queue number. Any bits that are set in EDMA_TPCC_CCERR also generate an EDMA_TPCC error interrupt.

16.2.5.11.4 Performance Considerations

The main system bus infrastructure (L3) arbitrates bus requests from all of the masters (TCs, CPU(S), and other bus masters) to the shared slave resources (peripherals and memories).

The priorities of transfer requests (read and write commands) from the EDMA transfer controllers with respect to other masters within the device IRQ_CROSSBAR are programmed using the Control Module registers. The EDMA_TPCC_QUEPRI register has no affect.

Therefore, the priority of unloading queues has a secondary affect compared to the priority of the transfers as they are executed by the EDMA_TPTC (dictated by the priority set using the Control Module registers, refer to Section 18.5 *Control Module Register Manual* in Chapter 18 *Control Module* chapter).

16.2.5.12 EDMA Transfer Controller (EDMA_TPTC)

The EDMA channel controller is the user-interface of the EDMA and the EDMA transfer controller (EDMA_TPTC) is the data movement engine of the EDMA controller. The EDMA_TPCC submits transfer requests (TR) to the EDMA_TPTC and the EDMA_TPTC performs the data transfers dictated by the TR, so the EDMA_TPTC is a slave to the EDMA_TPCC.

16.2.5.12.1 Architecture Details

16.2.5.12.1.1 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. An optimally-sized command is defined by the transfer controller default burst size (DBS), which is defined in Section 16.2.5.12.5 *EDMA_TPTC Configuration*.

The EDMA_TPTC attempts to issue the largest possible command size as limited by the DBS value or the EDMA_TPCC_ABCNT_n[15:0] ACNT and EDMA_TPCC_ABCNT_n[31:16] BCNT value of the TR. EDMA_TPTC 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 command always aligns the address of subsequent commands to the DBS value.

Table 16-105 lists the TR segmentation rules that are followed by the EDMA_TPTC. In summary, if the EDMA_TPCC_ABCNT_n[15:0] ACNT value is larger than the DBS value, then the EDMA_TPTC breaks the EDMA_TPCC_ABCNT_n[15:0] ACNT array into DBS-sized commands to the source/destination addresses. Each EDMA_TPCC_ABCNT_n[31:16] BCNT number of arrays are then serviced in succession.

For BCNT arrays of ACNT bytes (that is, a 2D transfer), if the EDMA_TPCC_ABCNT_n[15:0] ACNT value is less than or equal to the DBS value, then the TR may be optimized into a 1D-transfer in order to maximize efficiency. The optimization takes place if the EDMA_TPTC recognizes that the 2D-transfer is organized as a single dimension (EDMA_TPCC_ABCNT_n[15:0] ACNT == EDMA_TPCC_BIDX_n) and the ACNT value is a power of 2.

Table 16-105 lists conditions in which the optimizations are performed.

Table 16-105. Read/Write Command Optimization Rules

ACNT ≤ DBS	ACNT is power of 2	BIDX = ACNT	BCNT ≤ 1023	SAM/DAM = Increment	Description
Yes	Yes	Yes	Yes	Yes	Optimized
No	x	x	x	x	Not Optimized
x	No	x	x	x	Not Optimized
x	x	No	x	x	Not Optimized
x	x	x	No	x	Not Optimized
x	x	x	x	No	Not Optimized

16.2.5.12.1.2 TR Pipelining

TR pipelining refers to the ability of the source active set to proceed ahead of the destination active set. Essentially, the reads for a given TR may already be in progress while the writes of a previous TR may not have completed.

The number of outstanding TRs is limited by the number of destination FIFO register entries.

TR pipelining is useful for maintaining throughput on back-to-back small TRs. It minimizes the startup overhead because reads start in the background of a previous TR writes.

Example 16-4. Command Fragmentation (DBS = 64)

The pseudo code:

1. EDMA_TPTCn_PCNT[15:0] ACNT = 8, EDMA_TPTCn_PCNT[31:16] BCNT = 8, EDMA_TPTCn_PBIDX[15:0] SBIDX = 8, EDMA_TPTCn_PBIDX[31:16] DBIDX = 10, EDMA_TPTCn_PSRC[31:0] SADDR = 64, EDMA_TPTCn_SADST[31:0] DADDR = 191

   Read Controller: This is optimized from a 2D-transfer to a 1D-transfer such that the read side is equivalent to EDMA_TPTCn_PCNT[15:0] ACNT = 64, EDMA_TPTCn_PCNT[31:16] BCNT = 1.

   Cmd0 = 64 byte

   Write Controller: Because DBIDX != 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. EDMA_TPTCn_PCNT[15:0] ACNT=128, EDMA_TPTCn_PCNT[31:16] BCNT = 1, EDMA_TPTCn_PSRC[31:0] SADDR = 63, EDMA_TPTCn_SADST[31:0] DADDR = 513

   Read Controller: Read address is not aligned.

   Cmd0 = 1 byte, (now the SADDR is aligned to 64 for the next command)

   Cmd1 = 64 bytes

   Cmd2 = 63 bytes

   Write Controller: The write address is also not aligned.

   Cmd0 = 63 bytes, (now the DADDR is aligned to 64 for the next command)

   Cmd1 = 64 bytes

   Cmd2 = 1 byte

16.2.5.12.1.3 Performance Tuning

By default, reads are as issued as fast as possible. In some cases, the reads issued by the EDMA_TPTC 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 EDMA_TPTC is controlled by the EDMA_TPTCn_RDRATE register. The EDMA_TPTCn_RDRATE register defines the number of cycles that the EDMA_TPTC read controller waits before issuing subsequent commands for a given TR, thus minimizing the chance of the EDMA_TPTC consuming all available slave resources. The EDMA_TPTCn_RDRATE[2:0] RDRATE value must be set to a relatively small value if the transfer controller is targeted for high priority transfers and to a higher 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.

16.2.5.12.2 Memory Protection

The transfer controller plays an important role in handling proxy memory protection. There are two access properties associated with a transfer: for instance, the privilege id (system-wide identification assigned to a master) of the master initiating the transfer, and the privilege level (user versus supervisor) used to program the transfer. This information is maintained in the PaRAM set when it is programmed in the channel controller. When a TR is submitted to the transfer controller, this information is made available to the EDMA_TPTC and used by the EDMA_TPTC while issuing read and write commands. The read or write commands have the same privilege identification, and privilege level as that programmed in the EDMA transfer in the channel controller.

16.2.5.12.3 Error Generation

Errors are generated if enabled under three conditions:

- EDMA_TPTC detection of an error signaled by the source or destination address.
- Attempt to read or write to an invalid address in the configuration memory map.

- Detection of a constant addressing mode TR violating the constant addressing mode transfer rules (the source/destination addresses and source/destination indexes must be aligned to 32 bytes).

Either or all error types may be disabled. If an error bit is set and enabled, the error interrupt for the concerned transfer controller is generated.

16.2.5.12.4 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 EDMA_TPTC status register EDMA_TPTCn_TCSTAT has dedicated bit fields to indicate the ongoing activity within different parts of the transfer controller:

- The EDMA_TPTCn_TCSTAT[1] SRCACTV bit indicates whether the source active set is active.
- The EDMA_TPTCn_TCSTAT[6:4] DSTACTV bit indicates the number of TRs resident in the destination register active set at a given instance.
- The EDMA_TPTCn_TCSTAT[0] PROGBUSY bit indicates whether a valid TR is present in the DMA program set.

NOTE: If the TRs are in progression, it must realize that there is a chance that the values read from the EDMA_TPTC status registers will be inconsistent since the EDMA_TPTC changes the values of these registers due to ongoing activities.

It is recommended that to ensure no additional submission of TRs to the EDMA_TPTC in order to facilitate ease of debug.

16.2.5.12.4.1 Destination FIFO Register Pointer

The destination FIFO register pointer is implemented as a circular buffer with the start pointer being EDMA_TPTCn_TCSTAT[12:11] DFSTRTPTR and a buffer depth of usually 2 or 4. The EDMA_TPTC maintains two important status details in EDMA_TPTCn_TCSTAT that are used during advanced debugging, if necessary. The EDMA_TPTCn_TCSTAT[12:11] DFSTRTPTR is a start pointer, the index to the head of the destination FIFO register. The EDMA_TPTCn_TCSTAT[6:4] DSTACTV is a counter for the number of valid (occupied) entries. These registers are used to get a brief history of transfers.

Examples of some register field values and their interpretation:

- EDMA_TPTCn_TCSTAT[12:11] DFSTRTPTR = 0x0 and EDMA_TPTCn_TCSTAT[6:4] DSTACTV = 0x0 implies that no TRs are stored in the destination FIFO register.
- EDMA_TPTCn_TCSTAT[12:11] DFSTRTPTR = 0x1 and EDMA_TPTCn_TCSTAT[6:4] DSTACTV = 0x2 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.
- EDMA_TPTCn_TCSTAT[12:11] DFSTRTPTR = 0x3 and EDMA_TPTCn_TCSTAT[6:4] DSTACTV = 0x2 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.

16.2.5.12.5 EDMA_TPTC Configuration

Table 16-106 provides the configuration of the individual EDMA transfer controllers present on the device. The DBS for each transfer controller is defined by Control Module register (CTRL_CORE_CONTROL_IO_1) settings.

Table 16-106. EDMA Transfer Controller Configurations

Name	TC0	TC1
EDMA_TPTCn_TCCFG[2:0] FIFOSIZE	1024 bytes	1024 bytes
EDMA_TPTCn_TCCFG[5:4] BUSWIDTH	16 bytes	16 bytes
EDMA_TPTCn_TCCFG[9:8] DSTREGDEPTH	4 entries	4 entries

Table 16-106. EDMA Transfer Controller Configurations (continued)

Name	TC0	TC1
DBS	Defined by CTRL_CORE_CONTROL_IO_1[9:8] TC0_DEFAULT_BURST_SIZE	Defined by CTRL_CORE_CONTROL_IO_1[13:12] TC1_DEFAULT_BURST_SIZE

16.2.5.13 Event Dataflow

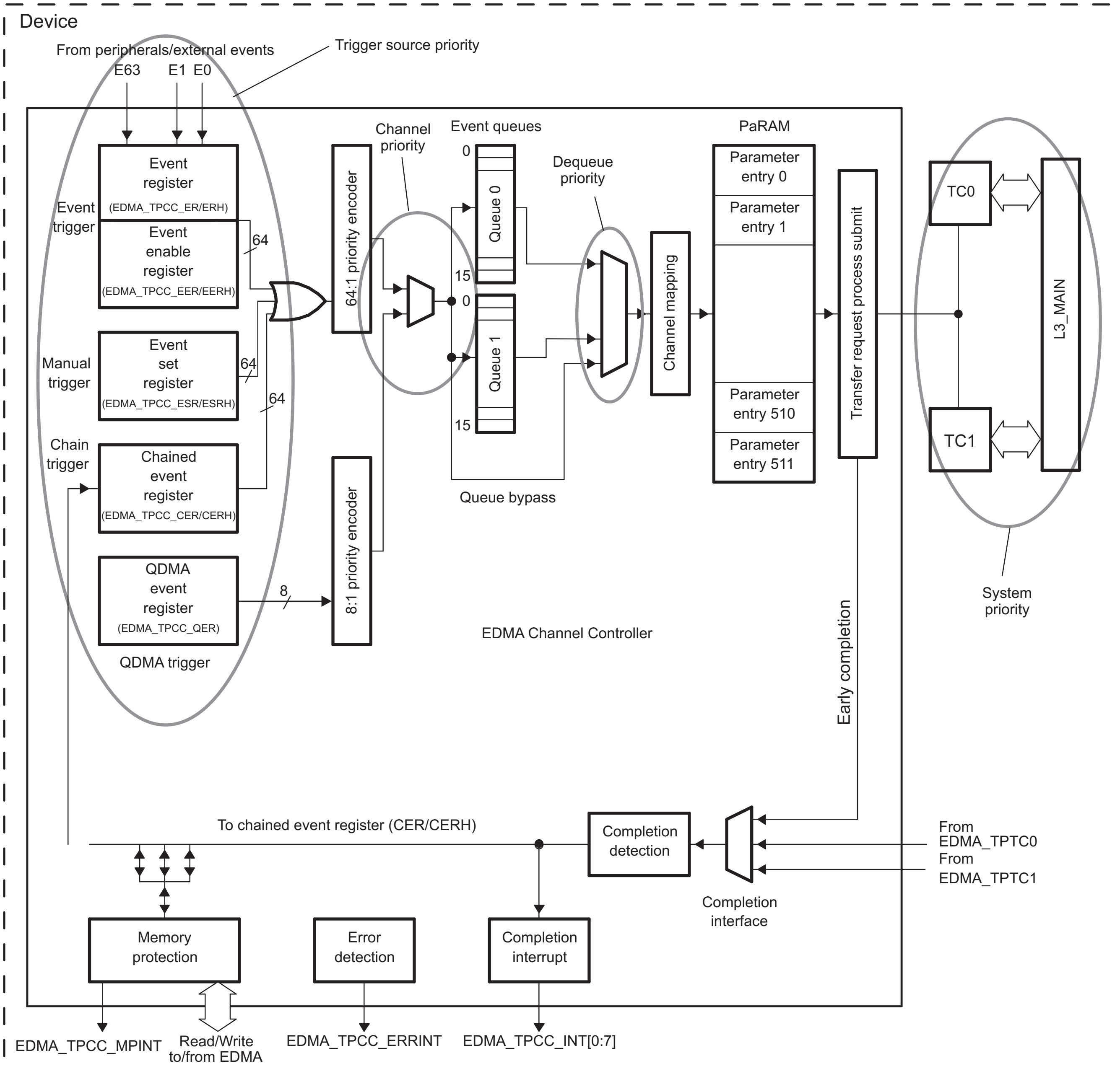
This section summarizes the data flow of a single event, from the time the event is latched to the channel controller to the time the transfer completion code is returned. The following steps list the sequence of EDMA_TPCC activity:

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 EDMA_TPCC_ER[31:0]E*n* / EDMA_TPCC_ERH[31:0] E*n* (or EDMA_TPCC_CER[31:0] E*n* / EDMA_TPCC_CERH[31:0] E*n*, EDMA_TPCC_ESR[31:0] E*n* / EDMA_TPCC_ESRH[31:0] E*n*, EDMA_TPCC_QER[7:0] E*n*) bit.
2. Once an event is prioritized and queued into the appropriate event queue, the EDMA_TPCC_SER[31:0] E*n* \ EDMA_TPCC_SERH[31:0] E*n* (or EDMA_TPCC_QSER[7:0] 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 EDMA_TPCC 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 EDMA_TPCC clears the EDMA_TPCC_ER[31:0] E*n*/ EDMA_TPCC_ERH[31:0] E*n* (or EDMA_TPCC_CER[31:0] E*n* / EDMA_TPCC_CERH[31:0] E*n*, EDMA_TPCC_ESR[31:0]E*n* / EDMA_TPCC_ESRH[31:0] E*n*, EDMA_TPCC_QER[31:0] E*n*) bit and the EDMA_TPCC_SER[31:0] E*n* / EDMA_TPCC_SERH[31:0] E*n* bit as soon as it determines the TR is non-null. In the case of a null set, the EDMA_TPCC_SER[31:0] E*n* / EDMA_TPCC_SERH[31:0] 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 EDMA_TPCC immediately sets the interrupt pending register (EDMA_TPCC_IPR[31:0] I[TCC] / EDMA_TPCC_IPRH[31:0] I[TCC] - 32).
5. If the TR was programmed for normal completion, the EDMA_TPCC sets the interrupt pending register (EDMA_TPCC_IPR[31:0] I[TCC] / EDMA_TPCC_IPRH[31:0] I[TCC]) when the EDMA_TPTC informs the EDMA_TPCC about completion of the transfer (returns transfer completion codes).
6. The EDMA_TPCC programs the associated EDMA_TPTC's 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 EDMA_TPTC*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 the EDMA_TPTC*n* then signals completion status to the EDMA_TPCC.

16.2.5.14 EDMA controller Prioritization

The EDMA 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 16-32 shows the different places EDMA priorities come into play.

Figure 16-32. EDMA Prioritization



16.2.5.14.1 Channel Priority

The EDMA event registers EDMA_TPCC_ER and EDMA_TPCC_ERH capture up to 64 events, the QDMA event register EDMA_TPCC_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 63 has the lowest priority, 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.

16.2.5.14.2 Trigger Source Priority

If a EDMA 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 (EDMA_TPCC_ER[31:0] E*n* = 1, EDMA_TPCC_ESR[31:0] E*n* = 1 , EDMA_TPCC_CER[31:0] E*n* = 1) , then the EDMA_TPCC always services these events in the following priority order: event trigger (via EDMA_TPCC_ER) is higher priority than chain trigger (via EDMA_TPCC_CER) and chain trigger is higher priority than manual trigger (via EDMA_TPCC_ESR).

This implies that if for channel 0, both EDMA_TPCC_ER[0] E0 = 1 and EDMA_TPCC_CER[0] E0 = 1 at the same time, then the EDMA_TPCC_ER[0] E0 event is always queued before the EDMA_TPCC_CER[0] E0 event.

16.2.5.14.3 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 EDMA_TPCC_DMAQNUMN_k and EDMA_TPCC_QDMAQNUM). For submission of a TR to the transfer request, events need to be de-queued from the event queues. Queue 0 has the highest dequeue priority and queue 1 the lowest.

16.2.5.15 EDMA Power, Reset and Clock Management

16.2.5.15.1 Clock and Power Management

The EDMA channel controller and transfer controller are clocked from L3MIAN1_L3_GICLK interface clock. The EDMA system runs at the L3 clock frequency.

The Auto clock gating for the EDMA_TPCC module is controled by the EDMA_TPCC_CLKGDIS[0] CLKGDIS bit at the module level.

The L3MIAN1_L3_GICLK interface clock to EDMA controller's modules are controled by the following registers in the PRCM module:

- PRCM.CM_L3MAIN1_TPCC_CLKCTRL - manages EDMA_TPCC module clock.
- PRCM.CM_L3MAIN1_TPTC1_CLKCTRL - manages EDMA_TPTC0 module clock.
- PRCM.CM_L3MAIN1_TPTC2_CLKCTRL - manages EDMA_TPTC1 module clock.

EDMA_TPCC and EDMA_TPTC0 and EDMA_TPTC1 modules have wakeup depandaces to several device modules. The wakeup dependency based on EDMA modules service requests are controled by registers in PRCM module:

- PRCM.PM_L3MAIN1_TPCC_WKDEP - controls wakeup dependency based on TPCC service requests.
- PRCM.PM_L3MAIN1_TPTC1_WKDEP - controls wakeup dependency based on TPTC0 service requests.
- PRCM.PM_L3MAIN1_TPTC2_WKDEP - controls wakeup dependency based on TPTC1 service requests.

  The EDMA_TPCC, EDMA_TPTC0 and EDMA_TPTC1 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 PRCM module. The PRCM acts as a master controller for power management for all peripherals on the device.

  The EDMA controller can be idled on receiving a clock stop request from the PRCM. The requests to EDMA_TPCC and EDMA_TPTC0 and EDMA_TPTC1 are separate. In general, it should be verified that there are no pending activities in the EDMA controller

NOTE: When EDMA controler modules no longer require the interface clock, software can disable it at the PRCM level by configuring the MODULEMODE bit field (PRCM.CM_L3MAIN1_TPCC_CLKCTRL[1:0], PRCM.CM_L3MAIN1_TPTC1_CLKCTRL[1:0], PRCM.CM_L3MAIN1_TPTC2_CLKCTRL[1:0]) in the PRCM registers. The clock is effectively cut, provided the other modules that receive it do not require it.

At the PRCM level, when all the conditions to shut off the L3MIAN1_L3_GICLK clock are met the PRCM module automatically launches a hardware handshake protocol to ensure EDMA modules are ready to have this clock switched off. Namely, the PRCM module asserts an IDLE request to the EDMA modules. For more information, refer to Chapter 3, *Power, Reset, and Clock Management*.

16.2.5.15.2 Reset Considerations

A hardware resets the EDMA (EDMA_TPCC, EDMA_TPTC0 and EDMA_TPTC1) and the EDMA configuration registers. The PaRAM memory contents are undefined after device reset and it should not rely on parameters to be reset to a known state. The PaRAM entry must be initialized to a desired value before it is used. The EDMA modules are reset by CORE_RET_RST reset signal from PRCM.

16.2.5.16 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 EDMA channel controller and transfer controller operations continue. Events continue to be latched and processed and transfer requests continue to be submitted and serviced.

Since EDMA is involved in servicing multiple master and slave peripherals, it is not feasible to have an independent behavior of the EDMA for emulation halts. EDMA functionality would be coupled with the peripherals it is servicing, which might have different behavior during emulation halts.

16.2.6 EDMA Transfer Examples

The EDMA channel controller performs a variety of transfers depending on the parameter configuration. The following sections provide a description and PaRAM configuration for some typical use case scenarios.

16.2.6.1 Block Move Example

The most basic transfer performed by the EDMA 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 as shown in Figure 16-33.

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 shown in Figure 16-34 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, EDMA_TPCC_ABCNT_n[31:16] BCNT and the B-indexes need to be set appropriately with the synchronization type set to AB-synchronized. The EDMA_TPCC_OPT_n[3] STATIC bit 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. The QDMA trigger word must be programed to be the highest numbered offset in the PaRAM set that undergoes change.

Figure 16-34 shows the parameters Block Move transfer.

Figure 16-33. Block Move Example

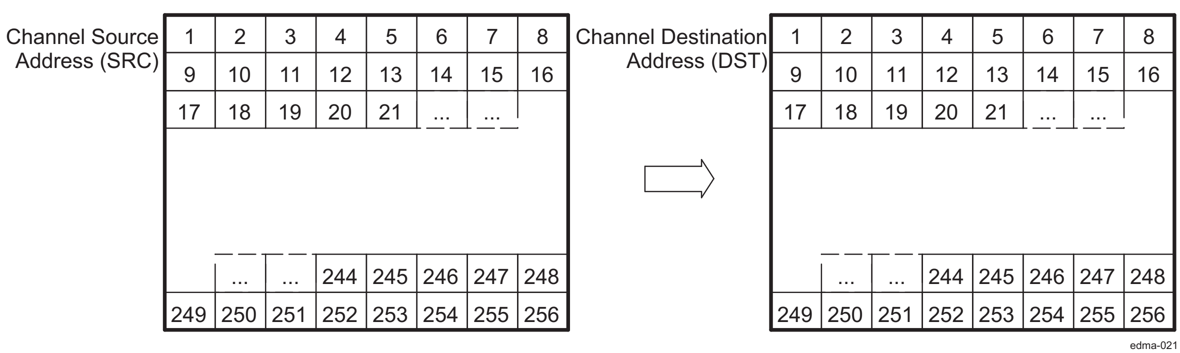


Figure 16-34. Block Move Example PaRAM Configuration

(a) EDMA Parameters

Parameter Contents		Parameter	
0010 0008h		Channel Options Parameter (OPT)	
Channel Source Address (SRC)		Channel Source Address (SRC)	
0001h	0100h	Count for 2nd Dimension (BCNT)	Count for 1st Dimension (ACNT)
Channel Destination Address (DST)		Channel Destination Address (DST)	
0000h	0000h	Destination BCNT Index (DBIDX)	Source BCNT Index (SBIDX)
0000h	FFFFh	BCNT Reload (BCNTRLD)	Link Address (LINK)
0000h	0000h	Destination CCNT Index (DCIDX)	Source CCNT Index (SCIDX)
0000h	0001h	Reserved	Count for 3rd Dimension (CCNT)

(b)Channel Options Parameter (OPT) Content

- EDMA_TPCC_OPT_n[3] STATIC = 0x1
- EDMA_TPCC_OPT_n[20] TCINTEN = 0x1

16.2.6.2 Subframe Extraction Example

The EDMA can efficiently extract a small frame of data from a larger frame of data. By performing a 2D-to-1D transfer, the EDMA 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. 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 EDMA places the subframe in internal L2 SRAM. Figure 16-35 shows the transfer of a subframe from external memory to L2.

The same PaRAM entry options are used for QDMA channels, as well as DMA channels. The EDMA_TPCC_OPT_n[3] STATIC bit is set to prevent linking. For successive transfers, only changed parameters need to be programmed before triggering the channel.

Figure 16-36 shows the parameters for Subframe Extraction transfer.

Figure 16-35. Subframe Extraction Transfer

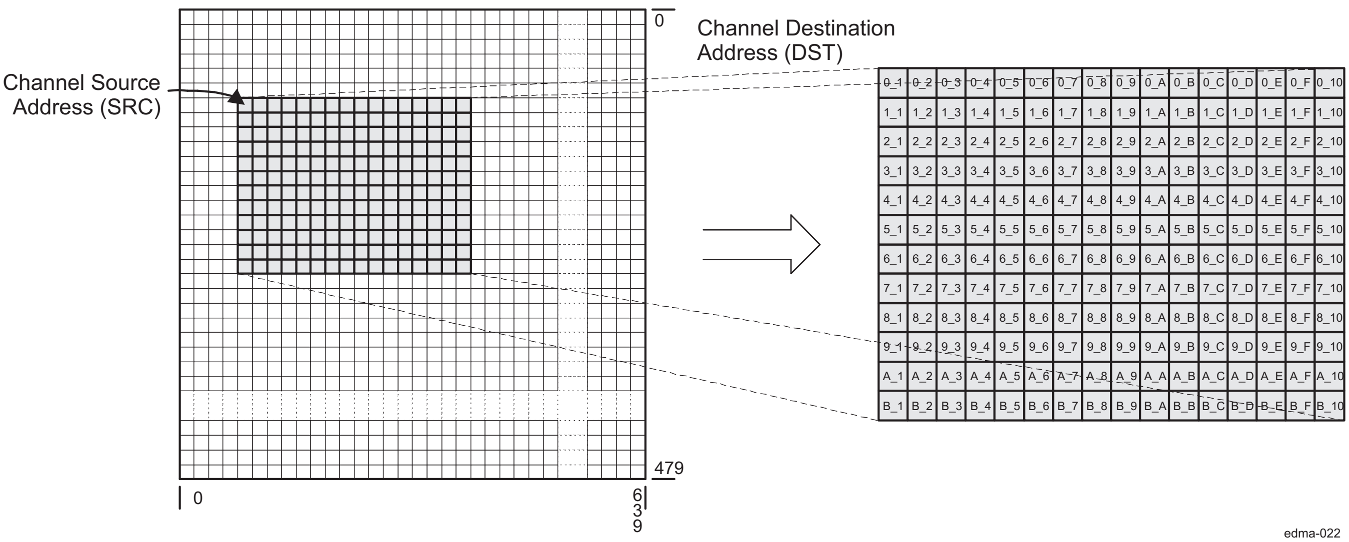


Figure 16-36. Subframe Extraction Example PaRAM Configuration

(a) EDMA Parameters

Parameter Contents	
0010 000Ch	
Channel Source Address (SRC)	
000Ch	0020h
Channel Destination Address (DST)	
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 (DBIDX)	Source BCNT Index (SBIDX)
BCNT Reload (BCNTRLD)	Link Address (LINK)
Destination CCNT Index (DCIDX)	Source CCNT Index (SCIDX)
Reserved	Count for 3rd Dimension (CCNT)

(b)Channel Options Parameter (OPT) Content

- EDMA_TPCC_OPT_n[2] SYNCDIM = 0x1
- EDMA_TPCC_OPT_n[3] STATIC = 0x1
- EDMA_TPCC_OPT_n[20] TCINTEN = 0x1

16.2.6.3 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 EDMA can reorganize the data into the desired format.

To determine the parameter set values, the following need to be considered:

- ACNT - Program this to be the size in bytes of an element.
- BCNT - Program this to be the number of elements in a frame.
- CCNT - Program this to be the number of frames.
- SBIDX - Program this to be the size of the element or ACNT.
- DBIDX - CCNT × ACNT
- SCIDX - ACNT × BCNT
- DCIDX - ACNT

The synchronization type needs to be AB-synchronized and the EDMA_TPCC_OPT_n[3] STATIC bit is 0 to allow updates to the parameter set. It is advised to use normal EDMA 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 elements get sorted, intermediate chaining could be used to trigger the channel again causing the transfer of the next BCNT elements and so on. Figure 16-38 shows the parameter set programming for this transfer, assuming channel 0 and an element size of 4 bytes.

Figure 16-37 shows the Data Sorting transfer

Figure 16-37. Data Sorting Example

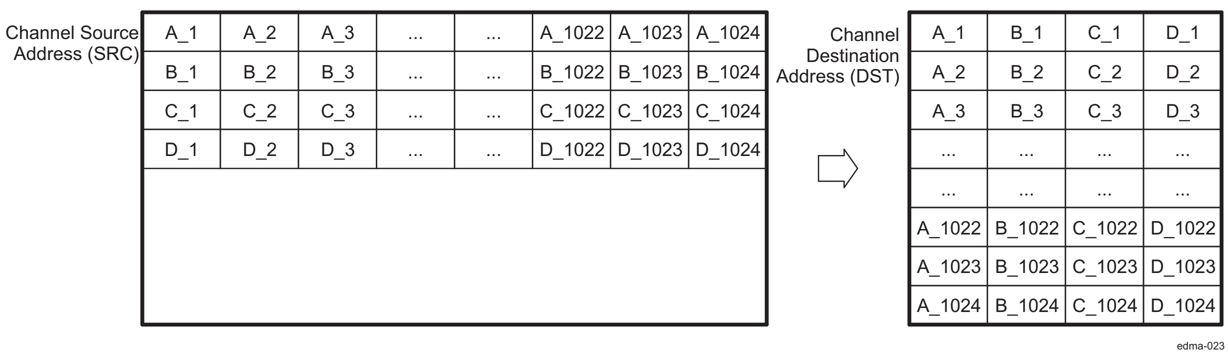


Figure 16-38. Data Sorting Example PaRAM Configuration

(a) EDMA Parameters

Parameter Contents		Parameter	
0090 0004h		Channel Options Parameter (OPT)	
Channel Source Address (SRC)		Channel Source Address (SRC)	
0400h	0004h	Count for 2nd Dimension (BCNT)	Count for 1st Dimension (ACNT)
Channel Destination Address (DST)		Channel Destination Address (DST)	
0010h	0001h	Destination BCNT Index (DSTBIDX)	Source BCNT Index (SRCBIDX)
0000h	FFFFh	BCNT Reload (BCNTRLD)	Link Address (LINK)
0001h	1000h	Destination CCNT Index (DSTCIDX)	Source CCNT Index (SRCCIDX)
0000h	0004h	Reserved	Count for 3rd Dimension (CCNT)

(b) Channel Options Parameter (OPT) Content

- EDMA_TPCC_OPT_n[2] SYNCDIM = 0x1
- EDMA_TPCC_OPT_n[20] TCINTEN = 0x1
- EDMA_TPCC_OPT_n[23] ITCCHEN = 0x1

16.2.6.4 Peripheral Servicing Example

The EDMA channel controller also services peripherals in the background of CPU operation, without requiring any CPU intervention. Through proper initialization of the EDMA channels, they can be configured to continuously service on-chip and off-chip peripherals throughout the device operation. Each event available to the EDMA 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 EDMA_TPCC_EER. When programming an EDMA channel to service a peripheral, it is necessary to know how data is to be presented to the processor. Data is always provided with some kind of synchronization event as either one element per event (non-bursting) or multiple elements per event (bursting).

16.2.6.4.1 Non-bursting Peripherals

Non-bursting peripherals include the on-chip multichannel audio serial port (McASP) and many external devices, such as codecs. Regardless of the peripheral, the EDMA channel configuration is the same.

The McASP transmit and receive data streams are treated independently by the EDMA. The transmit and receive data streams can have completely different counts, data sizes, and formats.

To transfer the incoming data stream to its proper location in DDR memory, the EDMA channel must be set up for a 1D-to-1D transfer with A-synchronization. Because an event (AREVT) is generated for every word as it arrives, it is necessary to have the EDMA issue the transfer request for each element individually. Figure 16-40 shows the parameters for this transfer. The source address of the EDMA channel is set to the data port address (DAT) for McASP, and the destination address is set to the start of the data block in DDR. Because the address of serializer buffer is fixed, the source B index is cleared to 0 (no modification) and the destination B index is set to 0x2 (increment).

Based on the premise that serial data is typically a high priority, the EDMA channel should be programmed to be on queue 0.

Figure 16-39 shows servicing incoming McASP data.

Figure 16-39. Servicing Incoming McASP Data Example

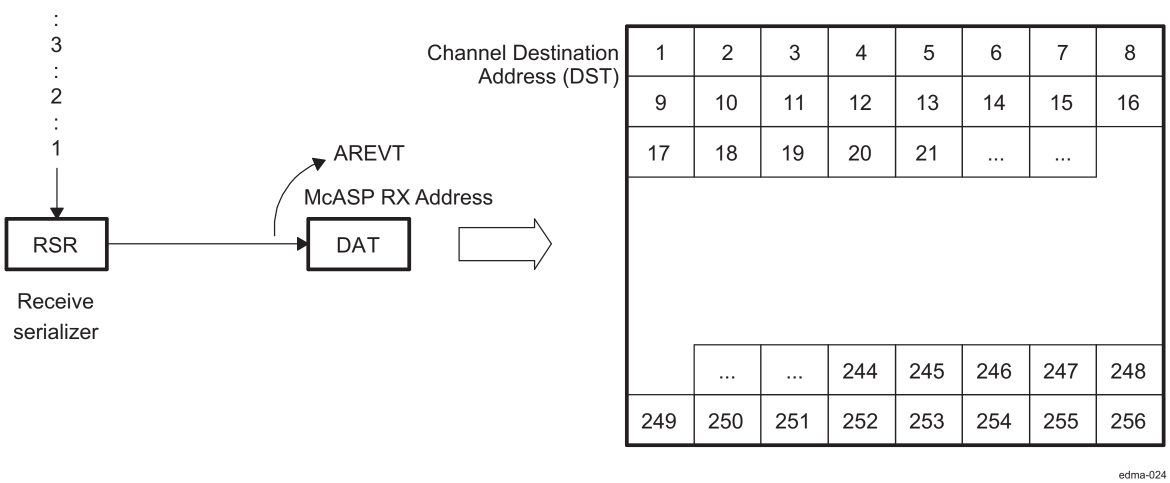


Figure 16-40. Servicing Incoming McASP Data Example PaRAM Configuration

(a) EDMA Parameters

Parameter Contents		Parameter	
0010 0000h		Channel Options Parameter (OPT)	
McASP RX Address		Channel Source Address (SRC)	
0100h	0001h	Count for 2nd Dimension (BCNT)	Count for 1st Dimension (ACNT)
Channel Destination Address (DST)		Channel Destination Address (DST)	
0001h	0000h	Destination BCNT Index (DBIDX)	Source BCNT Index (SBIDX)
0000h	FFFFh	BCNT Reload (BCNTRLD)	Link Address (LINK)
0000h	0000h	Destination CCNT Index (DCIDX)	Source CCNT Index (SCIDX)
0000h	0004h	Reserved	Count for 3rd Dimension (CCNT)

(b) Channel Options Parameter (OPT) Content

- EDMA_TPCC_OPT_n[20] TCINTEN = 0x1

16.2.6.4.2 Bursting Peripherals

Higher bandwidth applications require that multiple data elements be presented to the processor core 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 processor.

In this example, a port is receiving a video frame from a camera and presenting it to the DSP 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.

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 16-42 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. Because the input address is static, the EDMA_TPCC_BDIX_n[15:0] SBIDX is 0 (no modification to the source address). The destination is made up of arrays of contiguous, linear elements; therefore, the EDMA_TPCC_BIDX_n[31:16] DBIDX is set to pixel size, 2 bytes. EDMA_TPCC_ABCNT_n[15:0] ANCT is equal to the pixel size, 2 bytes. EDMA_TPCC_ABCNT_n[31:16] BCNT is set to the number of pixels in an array, 640. EDMA_TPCC_CCNT_n[15:0] CCNT is equal to the total number of arrays in the block, 480. EDMA_TPCC_CIDX_n[15:0] SCIDX is 0 because the source address undergoes no increment. The EDMA_TPCC_CIDX_n[31:16] DCIDX is equal to the difference between the starting addresses of each array. Because one pixel is 16 bits (2 bytes), EDMA_TPCC_CIDX_n[31:16] DCIDX is equal to 640 × 2.

Figure 16-41 shows Bursting Peripherals Transfer.

Figure 16-41. Servicing Peripheral Burst Example

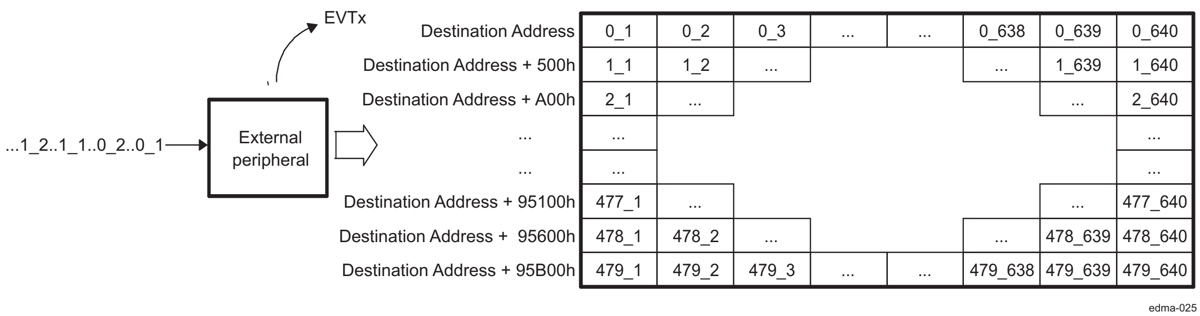


Figure 16-42. Servicing Peripheral Burst Example PaRAM Configuration

(a) EDMA Parameters

Parameter Contents	
0010 0004h	
Channel Source Address	
0280h	0002h
Channel Destination Address	
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

- EDMA_TPCC_OPT_n[2] SYNCDIM = 0x1
- EDMA_TPCC_OPT_n[20] TCINTEN = 0x1

16.2.6.4.3 Continuous Operation

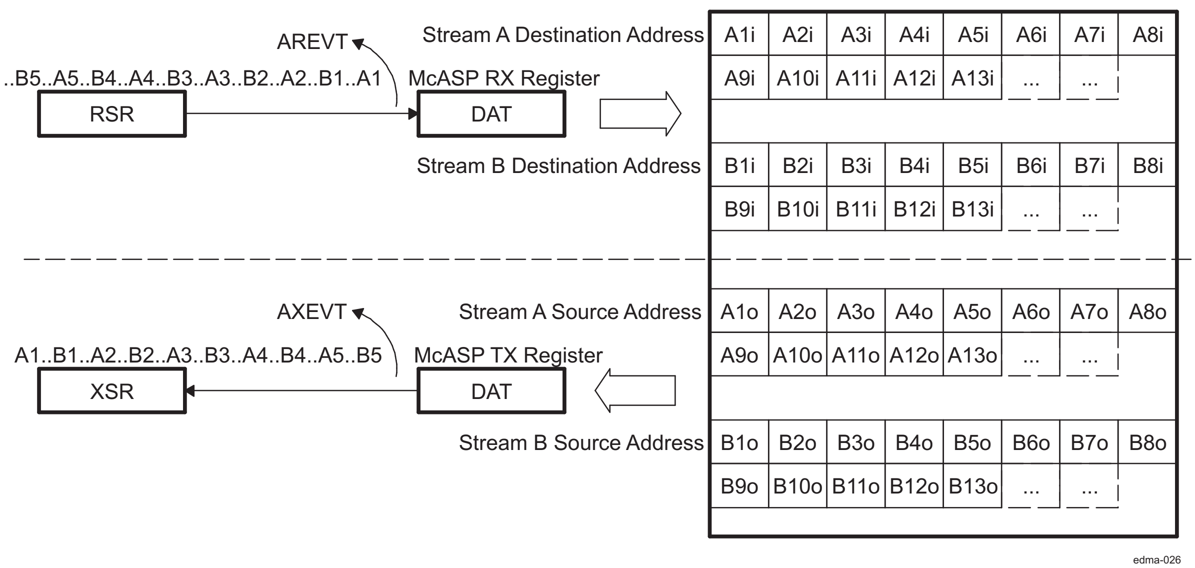
Configuring an EDMA channel to receive a single frame of data is useful, and is applicable to some systems. A majority of the time, however, data is going to be continuously transmitted and received throughout the entire operation of the processor. In this case, it is necessary to implement some form of linking such that the EDMA channels continuously reload the necessary parameter sets. In this example, McASP is configured to transmit and receive data on a T1 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 internal memory and from internal memory to the serial port, as shown in Figure 16-43.

The McASP generates AREVT for every element received and generates AXEVT for every element transmitted. To service the data streams, them DMA channels associated with the McASP must be setup for 1D-to-1D transfers with A-synchronization.

Figure 16-44 shows the parameter entries for the channel for these transfers. To service the McASP continuously throughout DSP operation, the channels must be linked to a duplicate PaRAM set in the PaRAM. After all frames have been transferred, the EDMA channels reload and continue.

Figure 16-45 shows the reload parameters for the channel.

Figure 16-43. Servicing Continuous McASP Data Example



16.2.6.4.3.1 Receive Channel

EDMA channel 15 services the incoming data stream of McASP. The source address is set to that of the receive serializer buffer, and the destination address is set to the first element of the data block. Because there are two data channels being serviced, A and B, they are to be located separately within the L2 SRAM.

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 15 parameter set, the parameters located at the link address are loaded into the channel 15 parameter set and operation continues. This function continues throughout device operation until halted by the CPU.

16.2.6.4.3.2 Transmit Channel

EDMA channel 12 services the outgoing data stream of McASP. 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 EDMA channel, with duplicate PaRAM set entries at PaRAM set 65.

Figure 16-44. Servicing Continuous McASP Data Example PaRAM Configuration

(a) EDMA Parameters for Receive Channel (PaRAM Set 15) being Linked to PaRAM Set 64

Parameter Contents		Parameter	
0010 0000h		Channel Options Parameter (OPT)	
McASP RX Register		Channel Source Address (SRC)	
0080h	0001h	Count for 2nd Dimension (BCNT)	Count for 1st Dimension (ACNT)
Channel Destination Address (DST)		Channel Destination Address (DST)	
0001h	0000h	Destination BCNT Index (DBIDX)	Source BCNT Index (SBIDX)
0080h	4800h	BCNT Reload (BCNTRLD)	Link Address (LINK)
0000h	0000h	Destination CCNT Index (DCIDX)	Source CCNT Index (SCIDX)
0000h	FFFFh	Reserved	Count for 3rd Dimension (CCNT)

(b) Channel Options Parameter (OPT) Content for Receive Channel (PaRAM Set 15)

- EDMA_TPCC_OPT_n[20] TCINTEN = 0x1

(c) EDMA Parameters for Transmit Channel (PaRAM Set 12) being Linked to PaRAM Set 65

Parameter Contents		Parameter	
0010 1000h		Channel Options Parameter (OPT)	
Channel Source Address (SRC)		Channel Source Address (SRC)	
0080h	0001h	Count for 2nd Dimension (BCNT)	Count for 1st Dimension (ACNT)
McASP TX Register		Channel Destination Address (DST)	
0000h	0001h	Destination BCNT Index (DBIDX)	Source BCNT Index (SBIDX)
0080h	4860h	BCNT Reload (BCNTRLD)	Link Address (LINK)
0000h	0000h	Destination CCNT Index (DCIDX)	Source CCNT Index (SCIDX)
0000h	FFFFh	Reserved	Count for 3rd Dimension (CCNT)

(d) Channel Options Parameter (OPT) Content for Transmit Channel (PaRAM Set 12)

- EDMA_TPCC_OPT_n[20] TCINTEN = 0x1

Figure 16-45. Servicing Continuous McASP Data Example Reload PaRAM Configuration

(a) EDMA Reload Parameters (PaRAM Set 64) for Receive Channel

Parameter Contents		Parameter	
0010 0000h		Channel Options Parameter (OPT)	
McASP RX Register		Channel Source Address (SRC)	
0080h	0001h	Count for 2nd Dimension (BCNT)	Count for 1st Dimension (ACNT)
Channel Destination Address (DST)		Channel Destination Address (DST)	
0001h	0000h	Destination BCNT Index (DBIDX)	Source BCNT Index (SBIDX)
0080h	4800h	BCNT Reload (BCNTRLD)	Link Address (LINK)
0000h	0000h	Destination CCNT Index (DCIDX)	Source CCNT Index (SCIDX)
0000h	FFFFh	Reserved	Count for 3rd Dimension (CCNT)

(b) Channel Options Parameter (OPT) Content for Receive Channel (PaRAM Set 64)

- EDMA_TPCC_OPT_n[20] TCINTEN = 0x1

(c) EDMA Reload Parameters (PaRAM Set 65) for Transmit Channel

Parameter Contents		Parameter	
0010 1000h		Channel Options Parameter (OPT)	
Channel Source Address (SRC)		Channel Source Address (SRC)	
0080h	0001h	Count for 2nd Dimension (BCNT)	Count for 1st Dimension (ACNT)
McASP TX Register		Channel Destination Address (DST)	
0000h	0001h	Destination BCNT Index (DBIDX)	Source BCNT Index (SBIDX)
0080h	4860h	BCNT Reload (BCNTRLD)	Link Address (LINK)
0000h	0000h	Destination CCNT Index (DCIDX)	Source CCNT Index (SCIDX)
0000h	FFFFh	Reserved	Count for 3rd Dimension (CCNT)

(d) Channel Options Parameter (OPT) Content for Transmit Channel (PaRAM Set 65)

- EDMA_TPCC_OPT_n[20] TCINTEN = 0x1

16.2.6.4.4 Ping-Pong Buffering

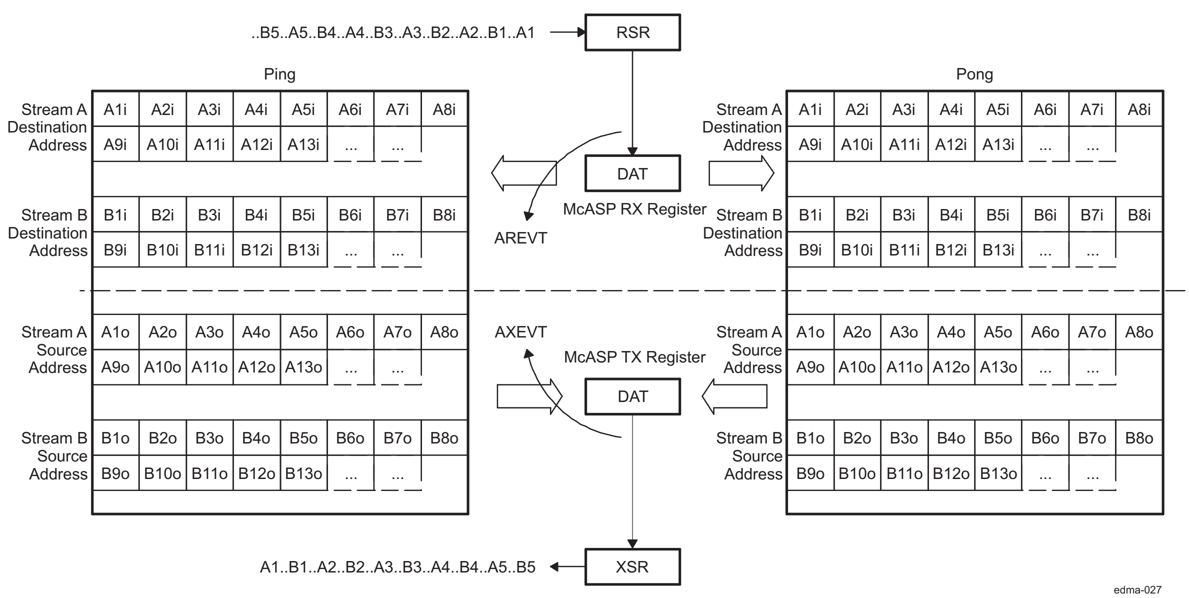
Although the previous configuration allows the EDMA to service a peripheral continuously, it presents a number of restrictions to the CPU. Because the input and output buffers are continuously being filled/emptied, the CPU must match the pace of the EDMA very closely to process the data. The EDMA receive data must always be placed in memory before the CPU accesses it, and the CPU must provide the output data before the EDMA 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 EDMA activity. This means that there are multiple (usually two) sets of data buffers for all incoming and outgoing data streams. While the EDMA transfers the data into and out of the ping buffers, the CPU manipulates the data in the pong buffers. When both CPU and EDMA activity completes, they switch. The EDMA then writes over the old input data and transfers the new output data. Figure 16-46 shows the ping-pong scheme for this example.

To change the continuous operation example, such that a ping-pong buffering scheme is used, the EDMA channels need only a moderate change. Instead of one 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 16-47 shows the EDMA channel configuration required.

Each channel has two parameter sets, ping and pong. The EDMA channel is initially loaded with the ping parameters (Figure 16-47). The link address for the ping set is set to the PaRAM offset of the pong parameter set (Figure 16-48). The link address for the pong set is set to the PaRAM offset of the ping parameter set (Figure 16-49). 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.

Figure 16-46. Ping-Pong Buffering for McASP Data Example



16.2.6.4.4.1 Synchronization with the CPU

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 EDMA 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 EDMA_TPCC_OPT_n[20] TCINTEN bit to generate an interrupt after completion. When channel 15 fills an input buffer, the E15 bit in the interrupt pending register

EDMA_TPCC_IPR is set; when channel 12 empties an output buffer, the E12 bit in EDMA_TPCC_IPR is set. The CPU must manually clear these bits. With the channel parameters set, the CPU polls EDMA_TPCC_IPR to determine when to switch. The EDMA 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 EDMA to complete.

Figure 16-47. Ping-Pong Buffering for McASP Example PaRAM Configuration

(a) EDMA Parameters for Channel 15 (Using PaRAM Set 15 Linked to Pong Set 64)

Parameter Contents		Parameter	
0010 D000h		Channel Options Parameter (OPT)	
McASP RX Register		Channel Source Address (SRC)	
0080h	0001h	Count for 2nd Dimension (BCNT)	Count for 1st Dimension (ACNT)
Channel Destination Address (DST)		Channel Destination Address (DST)	
0001h	0000h	Destination BCNT Index (DBIDX)	Source BCNT Index (SBIDX)
0080h	4800h	BCNT Reload (BCNTRLD)	Link Address (LINK)
0000h	0000h	Destination CCNT Index (DCIDX)	Source CCNT Index (SCIDX)
0000h	0001h	Reserved	Count for 3rd Dimension (CCNT)

(b) Channel Options Parameter (OPT) Content for Channel 15

- EDMA_TPCC_OPT_n[20] TCINTEN = 0x1

(c) EDMA Parameters for Channel 12 (Using PaRAM Set 12 Linked to Pong Set 65)

Parameter Contents		Parameter	
0010 C000h		Channel Options Parameter (OPT)	
Channel Source Address (SRC)		Channel Source Address (SRC)	
0080h	0001h	Count for 2nd Dimension (BCNT)	Count for 1st Dimension (ACNT)
McASP TX Register		Channel Destination Address (DST)	
0000h	0001h	Destination BCNT Index (DBIDX)	Source BCNT Index (SBIDX)
0080h	4840h	BCNT Reload (BCNTRLD)	Link Address (LINK)
0000h	0000h	Destination CCNT Index (DCIDX)	Source CCNT Index (SCIDX)
0000h	0001h	Reserved	Count for 3rd Dimension (CCNT)

(d) Channel Options Parameter (OPT) Content for Channel 12

- EDMA_TPCC_OPT_n[20] TCINTEN = 0x1

Figure 16-48. Ping-Pong Buffering for McASP Example Pong PaRAM Configuration

(a) EDMA Pong Parameters for Channel 15 at Set 64 Linked to Set 65

Parameter Contents

0010 D000h	
McASP RX Register	
0080h	0001h
Channel Destination Address (DST)	
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 (DBIDX)	Source BCNT Index (SBIDX)
BCNT Reload (BCNTRLD)	Link Address (LINK)
Destination CCNT Index (DCIDX)	Source CCNT Index (SCIDX)
Reserved	Count for 3rd Dimension (CCNT)

(b) EDMA Pong Parameters for Channel 12 at Set 66 Linked to Set 67

Parameter Contents

0010 C000h	
Channel Source Address (SRC)	
0080h	0001h
McASP TX Register	
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 (DBIDX)	Source BCNT Index (SBIDX)
BCNT Reload (BCNTRLD)	Link Address (LINK)
Destination CCNT Index (DCIDX)	Source CCNT Index (SCIDX)
Reserved	Count for 3rd Dimension (CCNT)

Figure 16-49. Ping-Pong Buffering for McASP Example Ping PaRAM Configuration

(a) EDMA Ping Parameters for Channel 15 at Set 65 Linked to Set 64

Parameter Contents

0010 D000h	
McASP RX Register	
0080h	0001h
Channel Destination Address (DST)	
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 (DBIDX)	Source BCNT Index (SBIDX)
BCNT Reload (BCNTRLD)	Link Address (LINK)
Destination CCNT Index (DCIDX)	Source CCNT Index (SCIDX)
Reserved	Count for 3rd Dimension (CCNT)

(b) EDMA Ping Parameters for Channel 12 at Set 67 Linked to Set 66

Parameter Contents

0010 C000h	
Channel Source Address (SRC)	
0080h	0001h
McASP TX Register	
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 (DBIDX)	Source BCNT Index (SBIDX)
BCNT Reload (BCNTRLD)	Link Address (LINK)
Destination CCNT Index (DCIDX)	Source CCNT Index (SCIDX)
Reserved	Count for 3rd Dimension (CCNT)

16.2.6.4.5 Transfer Chaining Examples

The following examples explain the intermediate transfer complete chaining function.

16.2.6.4.5.1 Servicing Input/Output FIFOs with a Single Event

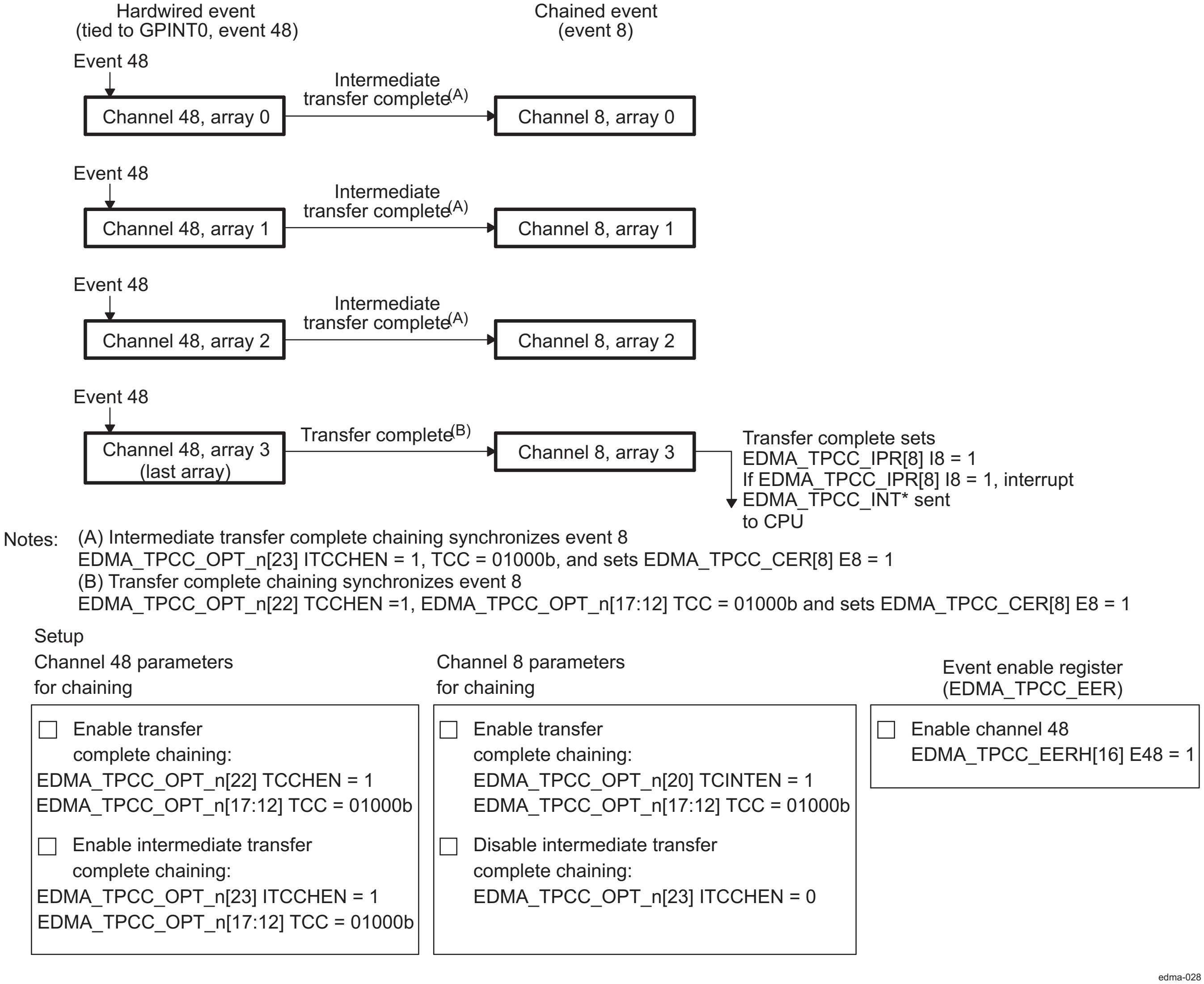
Many systems require the use of a pair of external FIFOs that must be serviced at the same rate. One FIFO buffers data input, and the other buffers data output. The EDMA channels that service these FIFOs can be set up for AB-synchronized transfers. While each FIFO is serviced with a different set of 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 EDMA 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 16-50 shows the EDMA 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 48, intermediate transfer complete chaining sets the E8 bit (specified by TCC of 8) in the chained event register EDMA_TPCC_CER and provides a synchronization event to channel 8. Upon completion of the last array transfer of channel 48, transfer complete chaining—not intermediate transfer complete chaining—sets the E8 bit in EDMA_TPCC_CER (specified by EDMA_TPCC_OPT_n[11] TCCMODE: TCC) and provides a synchronization event to channel 8. The completion of channel 8 sets the I8 bit (specified by EDMA_TPCC_OPT_n[11] TCCMODE: TCC) in the interrupt pending register EDMA_TPCC_IPR, which can generate an interrupt to the CPU, if the I8 bit in the interrupt enable register EDMA_TPCC_IER is set.

Figure 16-50 shows the Intermediate Transfer Completion Chaining Example.

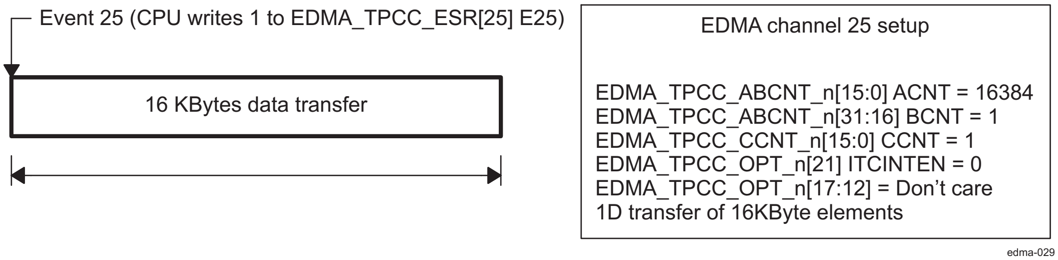
Figure 16-50. Intermediate Transfer Completion Chaining Example



16.2.6.4.5.2 *Breaking Up Large Transfers with Intermediate Chaining*

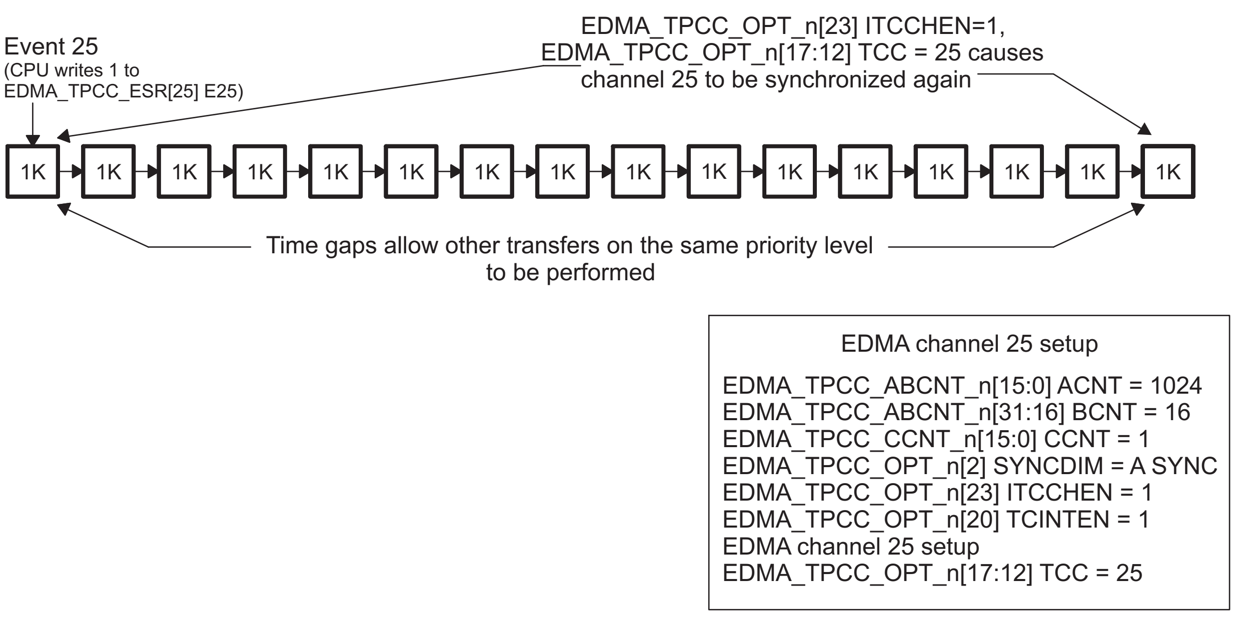
Another feature of intermediate transfer chaining EDMA_TPCC_OPT_n[23] ITCCHEN is for breaking up large transfers. A large transfer may lock out other transfers of the same priority level for the duration of the transfer. For example, a large transfer on queue 0 from the internal memory to the external memory using the EMIF may starve other EDMA 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 16-51 shows the EDMA setup and illustration of an example single large block transfer.

Figure 16-51. Single Large Block Transfer Example



The intermediate transfer chaining enable EDMA_TPCC_OPT_n[23] 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 EDMA 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 1 Kbyte is a reasonable small transfer in this example. The EDMA is set up to transfer 16 arrays of 1 Kbyte elements, for a total of 16K byte elements. The EDMA_TPCC_OPT_n[17:12] TCC field in the channel options parameter (OPT) is set to the same value as the channel number and EDMA_TPCC_OPT_n[23] ITCCHEN are set. In this example, EDMA channel 25 is used and EDMA_TPCC_OPT_n[17:12] TCC is also set to 25. The EDMA_TPCC_OPT_n[20] TCINTEN may also be set to trigger interrupt 25 when the last 1 Kbyte array is transferred. The CPU starts the EDMA transfer by writing to the appropriate bit of the event set register EDMA_TPCC_ESR[25] E25. The EDMA transfers the first 1 Kbyte array. Upon completion of the first array, intermediate transfer complete code chaining generates a synchronization event to channel 25, a value specified by the EDMA_TPCC_OPT_n[17:12] TCC field. This intermediate transfer completion chaining event causes EDMA channel 25 to transfer the next 1 Kbyte array. This process continues until the transfer parameters are exhausted, at which point the EDMA 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 16-52 shows the EDMA setup and illustration of the broken up smaller packet transfers.

Figure 16-52. Smaller Packet Data Transfers Example



16.2.6.5 Setting Up an EDMA Transfer

The following list provides a quick guide for the typical steps involved in setting up a transfer.

Step 1. Initiating a DMA/QDMA channel

(a) Determine the type of channel (QDMA or DMA) to be used.

(b) Channel mapping

(i) If using a QDMA channel, program the EDMA_TPCC_QCHMAPN_j with the parameter set number to which the channel maps and the trigger word.

(ii) If using a DMA channel, program the EDMA_TPCC_DCHMAPN_m with the parameter set number to which the channel maps.

(c) If the channel is being used in the context of a shadow region, ensure the EDMA_TPCC_DRAEM_k / EDMA_TPCC_DRAEHM_k for the region is properly set up to allow read write accesses to bits in the event registers and interrupt registers 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 16.2.5.7.1.)

(d) Determine the type of triggering used.

(i) If external events are used for triggering (DMA channels), enable the respective event in EDMA_TPCC_EER / EDMA_TPCC_EERH by writing into EDMA_TPCC_EESR / EDMA_TPCC_EESRH.

(ii) If QDMA Channel is used, enable the channel in EDMA_TPCC_QEER by writing into EDMA_TPCC_QEESR.

(e) Queue setup

(i) If a QDMA channel is used, set up the EDMA_TPCC_QDMAQNUM to map the channel to the respective event queue.

(ii) If a DMA channel is used, set up the EDMA_TPCC_DMAQNUMN_k to map the event to the respective event queue.

Step 2. Parameter set setup

(a) Program the PaRAM set number associated with the channel. Note that

---

NOTE: If it is a QDMA channel, the PaRAM entry that is configured as trigger word is written to last. Alternatively, enable the QDMA channel (step 1-b-ii above) just before the write to the trigger word.

Step 3. Interrupt setup

(a) Enable the interrupt in the EDMA_TPCC_IER / EDMA_TPCC_IERH by writing into EDMA_TPCC_IESR / EDMA_TPCC_IESRH.

(b) Ensure that the EDMA_TPCC completion interrupt (either the global or the shadow region interrupt) is enabled properly in the device interrupt controller.

(c) Ensure the EDMA_TPCC completion interrupt (this refers to either the Global interrupt or the shadow region interrupt) is enabled properly in the Device Interrupt controller.

(d) Set up the interrupt controller properly to receive the expected EDMA interrupt.

Step 4. Initiate transfer

(a) This step is highly dependent on the event trigger source:

(i) If the source is an external event coming from a peripheral, the peripheral will be enabled to start generating relevant EDMA events that can be latched to the EDMA_TPCC_ER transfer.

(ii) For QDMA events, writes to the trigger word (step 2-a above) will initiate the transfer.

(iii) Manually triggered transfers will be initiated by writes to the Event Set Registers EDMA_TPCC_ESR / EDMA_TPCC_ESRH.

(iv) Chained-trigger events initiate when a previous transfer returns a transfer completion code equal to the chained channel number.

Step 5. Wait for completion

(a) If the interrupts are enabled as mentioned in step 3 above, then the EDMA_TPCC will generate a completion interrupt to the CPU whenever transfer completion results in setting the corresponding bits in the interrupt pending register EDMA_TPCC_IPR / EDMA_TPCC_IPRH. The set bits must be cleared in the EDMA_TPCC_IPR / EDMA_TPCC_IPRH by writing to corresponding bit in EDMA_TPCC_ICR / EDMA_TPCC_ICRH.

(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 the EDMA_TPCC_IPR / EDMA_TPCC_IPRH. Again, the set bits in the EDMA_TPCC_IPR / EDMA_TPCC_IPRH must be manually cleared via EDMA_TPCC_ICR / EDMA_TPCC_ICRH before the next set of transfers is performed for the same transfer completion code values.

16.2.7 EDMA Debug Checklist and Programming Tips

This section lists some tips to keep in mind while debugging applications using the EDMA controller.

16.2.7.1 EDMA Debug Checklist

Table 16-107 provides some common issues and their probable causes and resolutions.

Table 16-107. Debug Checklist

Issue	Description/Solution
The transfer associated with the channel does not happen. The channel does not get serviced.	The EDMA_TPCC 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, i.e., if an external/peripheral event is latched in Event Registers EDMA_TPCC_ER / EDMA_TPCC_ERH, check that the event is enabled in the Event Enable Registers EDMA_TPCC_EER / EDMA_TPCC_EERH. Similarly, for QDMA channels, check that QDMA events are appropriately enabled in the QDMA Event Enable Register EDMA_TPCC_QEER. 2) Verify that the DMA or QDMA Secondary Event Register EDMA_TPCC_SER / EDMA_TPCC_SERH / EDMA_TPCC_QSER bits corresponding to the particular event or channel are not set.
The Secondary Event Registers 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 non-static and expected to be terminated by a NULL set (i.e., EDMA_TPCC_OPT_n[3] STATIC = 0x0, EDMA_TPCC_LNK_n[15:0] LINK = 0xFFFF), the parameter set is updated with a NULL set after submission of the last TR. Because QDMA channels are auto-triggered, 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 the EDMA_TPCC_QEMR and EDMA_TPCC_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 McASP, every time the data shifts out from the DXR register, 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 the EDMA_TPCC_SER[31:0] E*n* and EDMA_TPCC_EMR[31:0] E*n* set, preventing further event prioritization. Check the number of events received is limited to the expected number of events for which the parameter set is programmed, or check the bits corresponding to particular channel or event are not set in the Secondary event registers (EDMA_TPCC_SER / EDMA_TPCC_SERH / EDMA_TPCC_QSER) and Event Missed Registers (EDMA_TPCC_EMR / EDMA_TPCC_EMRH / EDMA_TPCC_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.	Check the following: 1) The interrupt generation is enabled in the EDMA_TPCC_OPT_n of the associated PaRAM set (EDMA_TPCC_OPT_n[20] TCINTEN = 0x1 and/or EDMA_TPCC_OPT_n[20] ITCINTEN = 0x1). 2) The interrupts are enabled in the EDMA Channel Controller, via the Interrupt Enable Registers (EDMA_TPCC_IER / EDMA_TPCC_IERH). 3) The corresponding interrupts are enabled in the device interrupt controller. 4) The set interrupts are cleared in the interrupt pending registers (EDMA_TPCC_IPR / EDMA_TPCC_IPRH) before exiting the transfer completion interrupt service routine (ISR). See Section 16.2.5.9.1.2 *Clearing Transfer Complerion Interrupts* for details on writing EDMA ISRs. 5) If working with shadow region interrupts, make sure that the DMA Region Access registers (EDMA_TPCC_DRAEM_k / EDMA_TPCC_DRAEHM_k) are set up properly, because the EDMA_TPCC_DRAEM_k / EDMA_TPCC_DRAEHM_k registers act as secondary enables for shadow region completion interrupts, along with the EDMA_TPCC_IER / EDMA_TPCC_IERH registers. If working with shadow region interrupts, make sure that the bits corresponding to the transfer completion code EDMA_TPCC_OPT_n[17:12] TCC value are also enabled in the EDMA_TPCC_DRAEM_k / EDMA_TPCC_DRAEHM_k registers. For instance, if the PaRAM set associated with Channel 0 returns a completion code of 63 EDMA_TPCC_OPT_n[17:12] TCC = 63, ensure that EDMA_TPCC_DRAEHM_k[31] E63 is also set for a shadow region completion interrupt because the interrupt pending register bit set will be EDMA_TPCC_IPRH[31] I63 (not EDMA_TPCC_IPR[0] I0).

16.2.7.2 EDMA Programming Tips

1. For several registers, the setting and clearing of bits needs to be done via separate dedicated registers.
   For example, the Event Register (EDMA_TPCC_ER / EDMA_TPCC_ERH) can only be cleared by writing a 1 to the corresponding bits in the Event Clear Registers (EDMA_TPCC_ECR / EDMA_TPCC_ECRH). Similarly, the Event Enable Register (EDMA_TPCC_EER / EDMA_TPCC_EERH) bits can only be set with writing of 0x1 to the Event Enable Set Registers (EDMA_TPCC_EESR / EDMA_TPCC_EESRH) and cleared with writing of 0x1 to the corresponding bits in the Event Enable Clear Register (EDMA_TPCC_EECR / EDMA_TPCC_EECRH).
2. Writes to the shadow region memory maps are governed by region access registers (EDMA_TPCC_DRAEM_k / EDMA_TPCC_DRAEHM_k / EDMA_TPCC_QRAEN_k). 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 Registers (EDMA_TPCC_DRAEM_k / EDMA_TPCC_DRAEHM_k) 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 EDMA_TPCC_DRAEM_k.DRAEM_0[0] E0 and EDMA_TPCC_DRAEM_k.DRAEM_1[0] E0 are both set, then on completion of a transfer that returns a TCC = 0x0, they will generate both shadow region 0 and 1 completion interrupts.
4. While programming a non-dummy parameter set, ensure the EDMA_TPCC_CCNT_n[15:0] CCNT is not left to zero.
5. Enable the EDMA_TPCC 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, it can 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, it chooses 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 EDMA_TPCC internally signals completion when the TR is submitted to the EDMA_TPTC, 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).

16.2.8 EDMA Register Manual

16.2.8.1 EDMA Instance Summary

Table 16-108shows the L3_MAIN base address and address space for the EDMA module instances.

Table 16-108. EDMA Instance Summary

Module Name	Base Address (L3_MAIN Access)	Size
SYS_EDMA_TPCC	0x4330 0000	1 MB
SYS_EDMA_TPTC0	0x4340 0000	1 MB
SYS_EDMA_TPTC1	0x4350 0000	1 MB
DSP1_EDMA_TPCC	0x40D1 0000	32 KB
DSP1_EDMA_TPTC0	0x40D0 5000	4 KB
DSP1_EDMA_TPTC1	0x40D0 6000	4 KB
DSP2_EDMA_TPCC	0x4151 0000	32 KB
DSP2_EDMA_TPTC0	0x4150 5000	4 KB
DSP2_EDMA_TPTC1	0x4150 6000	4 KB
EVE1_EDMA_TPCC(1)	0x420A 0000	32 KB
EVE2_EDMA_TPCC(1)	0x421A 0000	32 KB
EVE3_EDMA_TPCC(1)	0x422A 0000	32 KB
EVE4_EDMA_TPCC(1)	0x423A 0000	32 KB
EVE1_EDMA_TPTC0(1)	0x4208 6000	4 KB
EVE2_EDMA_TPTC0(1)	0x4218 6000	4 KB
EVE3_EDMA_TPTC0(1)	0x4228 6000	4 KB
EVE4_EDMA_TPTC0(1)	0x4238 6000	4 KB
EVE1_EDMA_TPTC1(1)	0x4208 7000	4 KB
EVE2_EDMA_TPTC1(1)	0x4218 7000	4 KB
EVE3_EDMA_TPTC1(1)	0x4228 7000	4 KB
EVE4_EDMA_TPTC1(1)	0x4238 7000	4 KB

(1) EVE is not supported in this family of devices.

Table 16-109 lists the base addresses for DSP internal (private) access to its embedded TPCC / TPTC modules.

Table 16-109. DSP Private Access EDMA Instance Summary

Module Name	Base Address	Size
DSP_EDMA_TPCC	0x01D1 0000	32 KB
DSP_EDMA_TPTC0	0x01D0 5000	4 KB
DSP_EDMA_TPTC1	0x01D0 6000	4 KB

Table 16-110 lists the base addresses for EVE internal (private) access to its embedded TPCC / TPTC modules.

Table 16-110. EVE EDMA Instance Summary (Private Access)

Module Name	Base Address (EVE Private Access)	Size
EVE_EDMA_TPCC	0x400A 0000	32 KB
EVE_EDMA_TPTC0	0x4008 6000	4 KB
EVE_EDMA_TPTC1	0x4008 7000	4 KB

16.2.8.2 EDMA Registers

16.2.8.2.1 EDMA Register Summary

Table 16-111 through Table 16-124 summarize the EDMA_TPCC, EDMA_TPTC0 and EDMA_TPTC1 registers.

NOTE: All EDMA modules in the SoC are functionally identical. Note that some of the configuration parameters may be different for the various EDMA instances (see Section 16.2.2, *EDMA Controllers Configuration*).

NOTE: EVE is not supported in this family of devices.

Table 16-111. System EDMA_TPCC Registers Mapping Summary

Register Name	Type	Register Width (Bits)	Address Offset	SYS_EDMA_TPCC Physical Address (L3_MAIN Access)
EDMA_TPCC_PID	R	32	0x0000 0000	0x4330 0000
EDMA_TPCC_CCCFG	R	32	0x0000 0004	0x4330 0004
EDMA_TPCC_CLKGDIS	RW	32	0x0000 00FC	0x4330 00FC
EDMA_TPCC_DCHMAPN_m [(1)]	RW	32	0x0000 0100 + (0x4 * m)	0x4330 0100 + (0x4 * m)
EDMA_TPCC_QCHMAPN_j [(2)]	RW	32	0x0000 0200 + (0x4 * j)	0x4330 0200 + (0x4 * j)
EDMA_TPCC_DMAQNUMN_k [(3)]	RW	32	0x0000 0240 + (0x4 * k)	0x4330 0240 + (0x4 * k)
EDMA_TPCC_QDMAQNUM	RW	32	0x0000 0260	0x4330 0260
EDMA_TPCC_QUETCMAP	RW	32	0x0000 0280	0x4330 0280
EDMA_TPCC_QUEPRI	RW	32	0x0000 0284	0x4330 0284
EDMA_TPCC_EMR	R	32	0x0000 0300	0x4330 0300
EDMA_TPCC_EMRH	R	32	0x0000 0304	0x4330 0304
EDMA_TPCC_EMCR	W	32	0x0000 0308	0x4330 0308
EDMA_TPCC_EMCRH	W	32	0x0000 030C	0x4330 030C
EDMA_TPCC_QEMR	R	32	0x0000 0310	0x4330 0310
EDMA_TPCC_QEMCR	W	32	0x0000 0314	0x4330 0314
EDMA_TPCC_CCERR	R	32	0x0000 0318	0x4330 0318
EDMA_TPCC_CCERRCLR	W	32	0x0000 031C	0x4330 031C
EDMA_TPCC_EEVAL	W	32	0x0000 0320	0x4330 0320
EDMA_TPCC_DRAEM_k [(3)]	RW	32	0x0000 0340 + (0x8 * k)	0x4330 0340 + (0x8 * k)
EDMA_TPCC_DRAEHM_k [(3)]	RW	32	0x0000 0344 + (0x8 * k)	0x4330 0344 + (0x8 * k)
EDMA_TPCC_QRAEN_k [(3)]	RW	32	0x0000 0380 + (0x4 * k)	0x4330 0380 + (0x4 * k)
EDMA_TPCC_Q0E_p [(4)]	R	32	0x0000 0400 + (0x4 *pl)	0x4330 0400 + (0x4 * p)
EDMA_TPCC_Q1E_p [(4)]	R	32	0x0000 0440 + (0x4 * p)	0x4330 0440 + (0x4 * p)
EDMA_TPCC_QSTATN_i [(5)]	R	32	0x0000 0600 + (0x4 * i)	0x4330 0600 + (0x4 * i)
EDMA_TPCC_QWMTHRA	RW	32	0x0000 0620	0x4330 0620
EDMA_TPCC_QWMTHRB	RW	32	0x0000 0624	0x4330 0624
EDMA_TPCC_CCSTAT	R	32	0x0000 0640	0x4330 0640
EDMA_TPCC_AETCTL	RW	32	0x0000 0700	0x4330 0700
EDMA_TPCC_AETSTAT	R	32	0x0000 0704	0x4330 0704
EDMA_TPCC_AETCMD	W	32	0x0000 0708	0x4330 0708

(1) m = 0 to 63 for SYS_EDMA_TPCC
(2) j = 0 to 7 for SYS_EDMA_TPCC
(3) k = 0 to 7 for SYS_EDMA_TPCC
(4) p = 0 to 15 for SYS_EDMA_TPCC
(5) i = 0 to 1 for SYS_EDMA_TPCC

Table 16-111. System EDMA_TPCC Registers Mapping Summary (continued)

Register Name	Type	Register Width (Bits)	Address Offset	SYS_EDMA_TPCC Physical Address (L3_MAIN Access)
EDMA_TPCC_MPFAR	R	32	0x0000 0800	0x4330 0800
EDMA_TPCC_MPFSR	R	32	0x0000 0804	0x4330 0804
EDMA_TPCC_MPFCR	W	32	0x0000 0808	0x4330 0808
EDMA_TPCC_MPPAG	RW	32	0x0000 080C	0x4330 080C
EDMA_TPCC_MPPAN_k [3]	RW	32	0x0000 0810 + (0x4 * k)	0x4330 0810 + (0x4 * k)
EDMA_TPCC_ER	R	32	0x0000 1000	0x4330 1000
EDMA_TPCC_ERH	R	32	0x0000 1004	0x4330 1004
EDMA_TPCC_ECR	W	32	0x0000 1008	0x4330 1008
EDMA_TPCC_ECRH	W	32	0x0000 100C	0x4330 100C
EDMA_TPCC_ESR	W	32	0x0000 1010	0x4330 1010
EDMA_TPCC_ESRH	W	32	0x0000 1014	0x4330 1014
EDMA_TPCC_CER	R	32	0x0000 1018	0x4330 1018
EDMA_TPCC_CERH	R	32	0x0000 101C	0x4330 101C
EDMA_TPCC_EER	R	32	0x0000 1020	0x4330 1020
EDMA_TPCC_EERH	R	32	0x0000 1024	0x4330 1024
EDMA_TPCC_EECR	W	32	0x0000 1028	0x4330 1028
EDMA_TPCC_EECRH	W	32	0x0000 102C	0x4330 102C
EDMA_TPCC_EESR	W	32	0x0000 1030	0x4330 1030
EDMA_TPCC_EESRH	W	32	0x0000 1034	0x4330 1034
EDMA_TPCC_SER	R	32	0x0000 1038	0x4330 1038
EDMA_TPCC_SERH	R	32	0x0000 103C	0x4330 103C
EDMA_TPCC_SECR	W	32	0x0000 1040	0x4330 1040
EDMA_TPCC_SECRH	W	32	0x0000 1044	0x4330 1044
EDMA_TPCC_IER	R	32	0x0000 1050	0x4330 1050
EDMA_TPCC_IERH	R	32	0x0000 1054	0x4330 1054
EDMA_TPCC_IECR	W	32	0x0000 1058	0x4330 1058
EDMA_TPCC_IECRH	W	32	0x0000 105C	0x4330 105C
EDMA_TPCC_IESR	W	32	0x0000 1060	0x4330 1060
EDMA_TPCC_IESRH	W	32	0x0000 1064	0x4330 1064
EDMA_TPCC_IPR	R	32	0x0000 1068	0x4330 1068
EDMA_TPCC_IPRH	R	32	0x0000 106C	0x4330 106C
EDMA_TPCC_ICR	W	32	0x0000 1070	0x4330 1070
EDMA_TPCC_ICRH	W	32	0x0000 1074	0x4330 1074
EDMA_TPCC_IEVAL	W	32	0x0000 1078	0x4330 1078
EDMA_TPCC_QER	R	32	0x0000 1080	0x4330 1080
EDMA_TPCC_QEER	R	32	0x0000 1084	0x4330 1084
EDMA_TPCC_QEECR	W	32	0x0000 1088	0x4330 1088
EDMA_TPCC_QEESR	W	32	0x0000 108C	0x4330 108C
EDMA_TPCC_QSER	R	32	0x0000 1090	0x4330 1090
EDMA_TPCC_QSECR	W	32	0x0000 1094	0x4330 1094
EDMA_TPCC_ER_RN_k [6]	R	32	0x0000 2000 + (0x200 * k)	0x4330 2000 + (0x200 * k)
EDMA_TPCC_ERH_RN_k [6]	R	32	0x0000 2004 + (0x200 * k)	0x4330 2004 + (0x200 * k)
EDMA_TPCC_ECR_RN_k [6]	W	32	0x0000 2008 + (0x200 * k)	0x4330 2008 + (0x200 * k)
EDMA_TPCC_ECRH_RN_k [6]	W	32	0x0000 200C + (0x200 * k)	0x4330 200C + (0x200 * k)
EDMA_TPCC_ESR_RN_k [6]	W	32	0x0000 2010 + (0x200 * k)	0x4330 2010 + (0x200 * k)

(6) k = 0 to 7 for SYS_EDMA_TPCC

Table 16-111. System EDMA_TPCC Registers Mapping Summary (continued)

Register Name	Type	Register Width (Bits)	Address Offset	SYS_EDMA_TPCC Physical Address (L3_MAIN Access)
EDMA_TPCC_ESRH_RN_k (6)	W	32	0x0000 2014 + (0x200 * k)	0x4330 2014 + (0x200 * k)
EDMA_TPCC_CER_RN_k (6)	R	32	0x0000 2018 + (0x200 * k)	0x4330 2018 + (0x200 * k)
EDMA_TPCC_CERH_RN_k (6)	R	32	0x0000 201C + (0x200 * k)	0x4330 201C + (0x200 * k)
EDMA_TPCC_EER_RN_k (6)	R	32	0x0000 2020 + (0x200 * k)	0x4330 2020 + (0x200 * k)
EDMA_TPCC_EERH_RN_k (6)	R	32	0x0000 2024 + (0x200 * k)	0x4330 2024 + (0x200 * k)
EDMA_TPCC_EECR_RN_k (6)	W	32	0x0000 2028 + (0x200 * k)	0x4330 2028 + (0x200 * k)
EDMA_TPCC_EECRH_RN_k (6)	W	32	0x0000 202C + (0x200 * k)	0x4330 202C + (0x200 * k)
EDMA_TPCC_EESR_RN_k (6)	W	32	0x0000 2030 + (0x200 * k)	0x4330 2030 + (0x200 * k)
EDMA_TPCC_EESRH_RN_k (6)	W	32	0x0000 2034 + (0x200 * k)	0x4330 2034 + (0x200 * k)
EDMA_TPCC_SER_RN_k (6)	R	32	0x0000 2038 + (0x200 * k)	0x4330 2038 + (0x200 * k)
EDMA_TPCC_SERH_RN_k (6)	R	32	0x0000 203C + (0x200 * k)	0x4330 203C + (0x200 * k)
EDMA_TPCC_SECR_RN_k (6)	W	32	0x0000 2040 + (0x200 * k)	0x4330 2040 + (0x200 * k)
EDMA_TPCC_SECRH_RN_k (6)	W	32	0x0000 2044 + (0x200 * k)	0x4330 2044 + (0x200 * k)
EDMA_TPCC_IER_RN_k (6)	R	32	0x0000 2050 + (0x200 * k)	0x4330 2050 + (0x200 * k)
EDMA_TPCC_IERH_RN_k (6)	R	32	0x0000 2054 + (0x200 * k)	0x4330 2054 + (0x200 * k)
EDMA_TPCC_IECR_RN_k (6)	W	32	0x0000 2058 + (0x200 * k)	0x4330 2058 + (0x200 * k)
EDMA_TPCC_IECRH_RN_k (6)	W	32	0x0000 205C + (0x200 * k)	0x4330 205C + (0x200 * k)
EDMA_TPCC_IESR_RN_k (6)	W	32	0x0000 2060 + (0x200 * k)	0x4330 2060 + (0x200 * k)
EDMA_TPCC_IESRH_RN_k (6)	W	32	0x0000 2064 + (0x200 * k)	0x4330 2064 + (0x200 * k)
EDMA_TPCC_IPR_RN_k (6)	R	32	0x0000 2068 + (0x200 * k)	0x4330 2068 + (0x200 * k)
EDMA_TPCC_IPRH_RN_k (6)	R	32	0x0000 206C + (0x200 * k)	0x4330 206C + (0x200 * k)
EDMA_TPCC_ICR_RN_k (6)	W	32	0x0000 2070 + (0x200 * k)	0x4330 2070 + (0x200 * k)
EDMA_TPCC_ICRH_RN_k (6)	W	32	0x0000 2074 + (0x200 * k)	0x4330 2074 + (0x200 * k)
EDMA_TPCC_IEVAL_RN_k (6)	W	32	0x0000 2078 + (0x200 * k)	0x4330 2078 + (0x200 * k)
EDMA_TPCC_QER_RN_k (6)	R	32	0x0000 2080 + (0x200 * k)	0x4330 2080 + (0x200 * k)
EDMA_TPCC_QEER_RN_k (7)	R	32	0x0000 2084 + (0x200 * k)	0x4330 2084 + (0x200 * k)
EDMA_TPCC_QEECR_RN_k (7)	W	32	0x0000 2088 + (0x200 * k)	0x4330 2088 + (0x200 * k)
EDMA_TPCC_QEESR_RN_k (7)	W	32	0x0000 208C + (0x200 * k)	0x4330 208C + (0x200 * k)
EDMA_TPCC_QSER_RN_k (7)	R	32	0x0000 2090 + (0x200 * k)	0x4330 2090 + (0x200 * k)
EDMA_TPCC_QSECR_RN_k (7)	W	32	0x0000 2094 + (0x200 * k)	0x4330 2094 + (0x200 * k)
EDMA_TPCC_OPT_n (8)	RW	32	0x0000 4000 + (0x20 * n)	0x4330 4000 + (0x20 * n)
EDMA_TPCC_SRC_n (8)	RW	32	0x0000 4004 + (0x20 * n)	0x4330 4004 + (0x20 * n)
EDMA_TPCC_ABCNT_n (8)	RW	32	0x0000 4008 + (0x20 * n)	0x4330 4008 + (0x20 * n)
EDMA_TPCC_DST_n (8)	RW	32	0x0000 400C + (0x20 * n)	0x4330 400C + (0x20 * n)
EDMA_TPCC_BIDX_n (8)	RW	32	0x0000 4010 + (0x20 * n)	0x4330 4010 + (0x20 * n)
EDMA_TPCC_LNK_n (8)	RW	32	0x0000 4014 + (0x20 * n)	0x4330 4014 + (0x20 * n)
EDMA_TPCC_CIDX_n (8)	RW	32	0x0000 4018 + (0x20 * n)	0x4330 4018 + (0x20 * n)
EDMA_TPCC_CCNT_n (8)	RW	32	0x0000 401C + (0x20 * n)	0x4330 401C + (0x20 * n)

(7) k = 0 to 7 for SYS_EDMA_TPCC
(8) n = 0 to 512 for SYS_EDMA_TPCC

Table 16-112. DSP1 EDMA_TPCC Registers Mapping Summary (L3_MAIN Access)

Register Name	Type	Register Width (Bits)	Address Offset	DSP1_EDMA_TPCC Physical Address (L3_MAIN Access)
EDMA_TPCC_PID	R	32	0x0000 0000	0x40D1 0000

Table 16-112. DSP1 EDMA_TPCC Registers Mapping Summary (L3_MAIN Access) (continued)

Register Name	Type	Register Width (Bits)	Address Offset	DSP1_EDMA_TPCC Physical Address (L3_MAIN Access)
EDMA_TPCC_CCCFG	R	32	0x0000 0004	0x40D1 0004
EDMA_TPCC_CLKGDIS	RW	32	0x0000 00FC	0x40D1 00FC
EDMA_TPCC_DCHMAPN_m [1]	RW	32	0x0000 0100 + (0x4 * m)	0x40D1 0100 + (0x4 * m)
EDMA_TPCC_QCHMAPN_j [2]	RW	32	0x0000 0200 + (0x4 * j)	0x40D1 0200 + (0x4 * j)
EDMA_TPCC_DMAQNUMN_k [3]	RW	32	0x0000 0240 + (0x4 * k)	0x40D1 0240 + (0x4 * k)
EDMA_TPCC_QDMAQNUM	RW	32	0x0000 0260	0x40D1 0260
EDMA_TPCC_QUETCMAP	RW	32	0x0000 0280	0x40D1 0280
EDMA_TPCC_QUEPRI	RW	32	0x0000 0284	0x40D1 0284
EDMA_TPCC_EMR	R	32	0x0000 0300	0x40D1 0300
EDMA_TPCC_EMRH	R	32	0x0000 0304	0x40D1 0304
EDMA_TPCC_EMCR	W	32	0x0000 0308	0x40D1 0308
EDMA_TPCC_EMCRH	W	32	0x0000 030C	0x40D1 030C
EDMA_TPCC_QEMR	R	32	0x0000 0310	0x40D1 0310
EDMA_TPCC_QEMCR	W	32	0x0000 0314	0x40D1 0314
EDMA_TPCC_CCERR	R	32	0x0000 0318	0x40D1 0318
EDMA_TPCC_CCERRCLR	W	32	0x0000 031C	0x40D1 031C
EDMA_TPCC_EEVAL	W	32	0x0000 0320	0x40D1 0320
EDMA_TPCC_DRAEM_k [3]	RW	32	0x0000 0340 + (0x8 * k)	0x40D1 0340 + (0x8 * k)
EDMA_TPCC_DRAEHM_k [3]	RW	32	0x0000 0344 + (0x8 * k)	0x40D1 0344 + (0x8 * k)
EDMA_TPCC_QRAEN_k [3]	RW	32	0x0000 0380 + (0x4 * k)	0x40D1 0380 + (0x4 * k)
EDMA_TPCC_Q0E_p [4]	R	32	0x0000 0400 + (0x4 *pl)	0x40D1 0400 + (0x4 * p)
EDMA_TPCC_Q1E_p [5]	R	32	0x0000 0440 + (0x4 * p)	0x40D1 0440 + (0x4 * p)
EDMA_TPCC_QSTATN_i [6]	R	32	0x0000 0600 + (0x4 * i)	0x40D1 0600 + (0x4 * i)
EDMA_TPCC_QWMTHRA	RW	32	0x0000 0620	0x40D1 0620
EDMA_TPCC_QWMTHRB	RW	32	0x0000 0624	0x40D1 0624
EDMA_TPCC_CCSTAT	R	32	0x0000 0640	0x40D1 0640
EDMA_TPCC_AETCTL	RW	32	0x0000 0700	0x40D1 0700
EDMA_TPCC_AETSTAT	R	32	0x0000 0704	0x40D1 0704
EDMA_TPCC_AETCMD	W	32	0x0000 0708	0x40D1 0708
EDMA_TPCC_MPFAR	R	32	0x0000 0800	0x40D1 0800
EDMA_TPCC_MPFSR	R	32	0x0000 0804	0x40D1 0804
EDMA_TPCC_MPFCR	W	32	0x0000 0808	0x40D1 0808
EDMA_TPCC_MPPAG	RW	32	0x0000 080C	0x40D1 080C
EDMA_TPCC_MPPAN_k [7]	RW	32	0x0000 0810 + (0x4 * k)	0x40D1 0810 + (0x4 * k)
EDMA_TPCC_ER	R	32	0x0000 1000	0x40D1 1000
EDMA_TPCC_ERH	R	32	0x0000 1004	0x40D1 1004
EDMA_TPCC_ECR	W	32	0x0000 1008	0x40D1 1008
EDMA_TPCC_ECRH	W	32	0x0000 100C	0x40D1 100C
EDMA_TPCC_ESR	W	32	0x0000 1010	0x40D1 1010
EDMA_TPCC_ESRH	W	32	0x0000 1014	0x40D1 1014

(1) m = 0 to 63 for DSP1_EDMA_TPCC
(2) j = 0 to 7 for DSP1_EDMA_TPCC
(3) k = 0 to 7 for DSP1_EDMA_TPCC
(4) p = 0 to 15 for DSP1_EDMA_TPCC
(5) p = 0 to 15 for DSP1_EDMA_TPCC
(6) i = 0 to 1 for DSP1_EDMA_TPCC
(7) k = 0 to 7 for DSP1_EDMA_TPCC

Table 16-112. DSP1 EDMA_TPCC Registers Mapping Summary (L3_MAIN Access) (continued)

Register Name	Type	Register Width (Bits)	Address Offset	DSP1_EDMA_TPCC Physical Address (L3_MAIN Access)
EDMA_TPCC_CER	R	32	0x0000 1018	0x40D1 1018
EDMA_TPCC_CERH	R	32	0x0000 101C	0x40D1 101C
EDMA_TPCC_EER	R	32	0x0000 1020	0x40D1 1020
EDMA_TPCC_EERH	R	32	0x0000 1024	0x40D1 1024
EDMA_TPCC_EECR	W	32	0x0000 1028	0x40D1 1028
EDMA_TPCC_EECRH	W	32	0x0000 102C	0x40D1 102C
EDMA_TPCC_EESR	W	32	0x0000 1030	0x40D1 1030
EDMA_TPCC_EESRH	W	32	0x0000 1034	0x40D1 1034
EDMA_TPCC_SER	R	32	0x0000 1038	0x40D1 1038
EDMA_TPCC_SERH	R	32	0x0000 103C	0x40D1 103C
EDMA_TPCC_SECR	W	32	0x0000 1040	0x40D1 1040
EDMA_TPCC_SECRH	W	32	0x0000 1044	0x40D1 1044
EDMA_TPCC_IER	R	32	0x0000 1050	0x40D1 1050
EDMA_TPCC_IERH	R	32	0x0000 1054	0x40D1 1054
EDMA_TPCC_IECR	W	32	0x0000 1058	0x40D1 1058
EDMA_TPCC_IECRH	W	32	0x0000 105C	0x40D1 105C
EDMA_TPCC_IESR	W	32	0x0000 1060	0x40D1 1060
EDMA_TPCC_IESRH	W	32	0x0000 1064	0x40D1 1064
EDMA_TPCC_IPR	R	32	0x0000 1068	0x40D1 1068
EDMA_TPCC_IPRH	R	32	0x0000 106C	0x40D1 106C
EDMA_TPCC_ICR	W	32	0x0000 1070	0x40D1 1070
EDMA_TPCC_ICRH	W	32	0x0000 1074	0x40D1 1074
EDMA_TPCC_IEVAL	W	32	0x0000 1078	0x40D1 1078
EDMA_TPCC_QER	R	32	0x0000 1080	0x40D1 1080
EDMA_TPCC_QEER	R	32	0x0000 1084	0x40D1 1084
EDMA_TPCC_QEECR	W	32	0x0000 1088	0x40D1 1088
EDMA_TPCC_QEESR	W	32	0x0000 108C	0x40D1 108C
EDMA_TPCC_QSER	R	32	0x0000 1090	0x40D1 1090
EDMA_TPCC_QSECR	W	32	0x0000 1094	0x40D1 1094
EDMA_TPCC_ER_RN_k [7]	R	32	0x0000 2000 + (0x200 * k)	0x40D1 2000 + (0x200 * k)
EDMA_TPCC_ERH_RN_k [8]	R	32	0x0000 2004 + (0x200 * k)	0x40D1 2004 + (0x200 * k)
EDMA_TPCC_ECR_RN_k [8]	W	32	0x0000 2008 + (0x200 * k)	0x40D1 2008 + (0x200 * k)
EDMA_TPCC_ECRH_RN_k [8]	W	32	0x0000 200C + (0x200 * k)	0x40D1 200C + (0x200 * k)
EDMA_TPCC_ESR_RN_k [8]	W	32	0x0000 2010 + (0x200 * k)	0x40D1 2010 + (0x200 * k)
EDMA_TPCC_ESRH_RN_k [8]	W	32	0x0000 2014 + (0x200 * k)	0x40D1 2014 + (0x200 * k)
EDMA_TPCC_CER_RN_k [8]	R	32	0x0000 2018 + (0x200 * k)	0x40D1 2018 + (0x200 * k)
EDMA_TPCC_CERH_RN_k [8]	R	32	0x0000 201C + (0x200 * k)	0x40D1 201C + (0x200 * k)
EDMA_TPCC_EER_RN_k [8]	R	32	0x0000 2020 + (0x200 * k)	0x40D1 2020 + (0x200 * k)
EDMA_TPCC_EERH_RN_k [8]	R	32	0x0000 2024 + (0x200 * k)	0x40D1 2024 + (0x200 * k)
EDMA_TPCC_EECR_RN_k [8]	W	32	0x0000 2028 + (0x200 * k)	0x40D1 2028 + (0x200 * k)
EDMA_TPCC_EECRH_RN_k [8]	W	32	0x0000 202C + (0x200 * k)	0x40D1 202C + (0x200 * k)
EDMA_TPCC_EESR_RN_k [8]	W	32	0x0000 2030 + (0x200 * k)	0x40D1 2030 + (0x200 * k)
EDMA_TPCC_EESRH_RN_k [8]	W	32	0x0000 2034 + (0x200 * k)	0x40D1 2034 + (0x200 * k)
EDMA_TPCC_SER_RN_k [8]	R	32	0x0000 2038 + (0x200 * k)	0x40D1 2038 + (0x200 * k)
EDMA_TPCC_SERH_RN_k [8]	R	32	0x0000 203C + (0x200 * k)	0x40D1 203C + (0x200 * k)

(8) k = 0 to 7 for DSP1_EDMA_TPCC

Table 16-112. DSP1 EDMA_TPCC Registers Mapping Summary (L3_MAIN Access) (continued)

Register Name	Type	Register Width (Bits)	Address Offset	DSP1_EDMA_TPCC Physical Address (L3_MAIN Access)
EDMA_TPCC_SECR_RN_k [8]	W	32	0x0000 2040 + (0x200 * k)	0x40D1 2040 + (0x200 * k)
EDMA_TPCC_SECRH_RN_k [8]	W	32	0x0000 2044 + (0x200 * k)	0x40D1 2044 + (0x200 * k)
EDMA_TPCC_IER_RN_k [8]	R	32	0x0000 2050 + (0x200 * k)	0x40D1 2050 + (0x200 * k)
EDMA_TPCC_IERH_RN_k [8]	R	32	0x0000 2054 + (0x200 * k)	0x40D1 2054 + (0x200 * k)
EDMA_TPCC_IECR_RN_k [8]	W	32	0x0000 2058 + (0x200 * k)	0x40D1 2058 + (0x200 * k)
EDMA_TPCC_IECRH_RN_k [8]	W	32	0x0000 205C + (0x200 * k)	0x40D1 205C + (0x200 * k)
EDMA_TPCC_IESR_RN_k [8]	W	32	0x0000 2060 + (0x200 * k)	0x40D1 2060 + (0x200 * k)
EDMA_TPCC_IESRH_RN_k [8]	W	32	0x0000 2064 + (0x200 * k)	0x40D1 2064 + (0x200 * k)
EDMA_TPCC_IPR_RN_k [8]	R	32	0x0000 2068 + (0x200 * k)	0x40D1 2068 + (0x200 * k)
EDMA_TPCC_IPRH_RN_k [8]	R	32	0x0000 206C + (0x200 * k)	0x40D1 206C + (0x200 * k)
EDMA_TPCC_ICR_RN_k [8]	W	32	0x0000 2070 + (0x200 * k)	0x40D1 2070 + (0x200 * k)
EDMA_TPCC_ICRH_RN_k [8]	W	32	0x0000 2074 + (0x200 * k)	0x40D1 2074 + (0x200 * k)
EDMA_TPCC_IEVAL_RN_k [8]	W	32	0x0000 2078 + (0x200 * k)	0x40D1 2078 + (0x200 * k)
EDMA_TPCC_QER_RN_k [8]	R	32	0x0000 2080 + (0x200 * k)	0x40D1 2080 + (0x200 * k)
EDMA_TPCC_QEER_RN_k [8]	R	32	0x0000 2084 + (0x200 * k)	0x40D1 2084 + (0x200 * k)
EDMA_TPCC_QEECR_RN_k [8]	W	32	0x0000 2088 + (0x200 * k)	0x40D1 2088 + (0x200 * k)
EDMA_TPCC_QEESR_RN_k [8]	W	32	0x0000 208C + (0x200 * k)	0x40D1 208C + (0x200 * k)
EDMA_TPCC_QSER_RN_k [8]	R	32	0x0000 2090 + (0x200 * k)	0x40D1 2090 + (0x200 * k)
EDMA_TPCC_QSECR_RN_k [8]	W	32	0x0000 2094 + (0x200 * k)	0x40D1 2094 + (0x200 * k)
EDMA_TPCC_OPT_n [9]	RW	32	0x0000 4000 + (0x20 * n)	0x40D1 4000 + (0x20 * n)
EDMA_TPCC_SRC_n [9]	RW	32	0x0000 4004 + (0x20 * n)	0x40D1 4004 + (0x20 * n)
EDMA_TPCC_ABCNT_n [9]	RW	32	0x0000 4008 + (0x20 * n)	0x40D1 4008 + (0x20 * n)
EDMA_TPCC_DST_n [9]	RW	32	0x0000 400C + (0x20 * n)	0x40D1 400C + (0x20 * n)
EDMA_TPCC_BIDX_n [9]	RW	32	0x0000 4010 + (0x20 * n)	0x40D1 4010 + (0x20 * n)
EDMA_TPCC_LNK_n [9]	RW	32	0x0000 4014 + (0x20 * n)	0x40D1 4014 + (0x20 * n)
EDMA_TPCC_CIDX_n [10]	RW	32	0x0000 4018 + (0x20 * n)	0x40D1 4018 + (0x20 * n)
EDMA_TPCC_CCNT_n [10]	RW	32	0x0000 401C + (0x20 * n)	0x40D1 401C + (0x20 * n)

(9) n = 0 to 127 for DSP1_EDMA_TPCC
(10) n = 0 to 127 for DSP1_EDMA_TPCC

Table 16-113. DSP2 EDMA_TPCC Registers Mapping Summary (L3_MAIN Access)

Register Name	Type	Register Width (Bits)	Address Offset	DSP2_EDMA_TPCC Physical Address (L3_MAIN Access)
EDMA_TPCC_PID	R	32	0x0000 0000	0x4151 0000
EDMA_TPCC_CCCFG	R	32	0x0000 0004	0x4151 0004
EDMA_TPCC_CLKGDIS	RW	32	0x0000 00FC	0x4151 00FC
EDMA_TPCC_DCHMAPN_m [1]	RW	32	0x0000 0100 + (0x4 * m)	0x4151 0100 + (0x4 * m)
EDMA_TPCC_QCHMAPN_j [2]	RW	32	0x0000 0200 + (0x4 * j)	0x4151 0200 + (0x4 * j)
EDMA_TPCC_DMAQNUMN_k [3]	RW	32	0x0000 0240 + (0x4 * k)	0x4151 0240 + (0x4 * k)
EDMA_TPCC_QDMAQNUM	RW	32	0x0000 0260	0x4151 0260
EDMA_TPCC_QUETCMAP	RW	32	0x0000 0280	0x4151 0280
EDMA_TPCC_QUEPRI	RW	32	0x0000 0284	0x4151 0284
EDMA_TPCC_EMR	R	32	0x0000 0300	0x4151 0300

(1) m = 0 to 63 for DSP2_EDMA_TPCC
(2) j = 0 to 7 for DSP2_EDMA_TPCC
(3) k = 0 to 7 for DSP2_EDMA_TPCC

Table 16-113. DSP2 EDMA_TPCC Registers Mapping Summary (L3_MAIN Access) (continued)

Register Name	Type	Register Width (Bits)	Address Offset	DSP2_EDMA_TPCC Physical Address (L3_MAIN Access)
EDMA_TPCC_EMRH	R	32	0x0000 0304	0x4151 0304
EDMA_TPCC_EMCR	W	32	0x0000 0308	0x4151 0308
EDMA_TPCC_EMCRH	W	32	0x0000 030C	0x4151 030C
EDMA_TPCC_QEMR	R	32	0x0000 0310	0x4151 0310
EDMA_TPCC_QEMCR	W	32	0x0000 0314	0x4151 0314
EDMA_TPCC_CCERR	R	32	0x0000 0318	0x4151 0318
EDMA_TPCC_CCERRCLR	W	32	0x0000 031C	0x4151 031C
EDMA_TPCC_EEVAL	W	32	0x0000 0320	0x4151 0320
EDMA_TPCC_DRAEM_k [3]	RW	32	0x0000 0340 + (0x8 * k)	0x4151 0340 + (0x8 * k)
EDMA_TPCC_DRAEHM_k [3]	RW	32	0x0000 0344 + (0x8 * k)	0x4151 0344 + (0x8 * k)
EDMA_TPCC_QRAEN_k [3]	RW	32	0x0000 0380 + (0x4 * k)	0x4151 0380 + (0x4 * k)
EDMA_TPCC_Q0E_p [4]	R	32	0x0000 0400 + (0x4 * p)	0x4151 0400 + (0x4 * p)
EDMA_TPCC_Q1E_p [4]	R	32	0x0000 0440 + (0x4 * p)	0x4151 0440 + (0x4 * p)
EDMA_TPCC_QSTATN_i [5]	R	32	0x0000 0600 + (0x4 * i)	0x4151 0600 + (0x4 * i)
EDMA_TPCC_QWMTHRA	RW	32	0x0000 0620	0x4151 0620
EDMA_TPCC_QWMTHRB	RW	32	0x0000 0624	0x4151 0624
EDMA_TPCC_CCSTAT	R	32	0x0000 0640	0x4151 0640
EDMA_TPCC_AETCTL	RW	32	0x0000 0700	0x4151 0700
EDMA_TPCC_AETSTAT	R	32	0x0000 0704	0x4151 0704
EDMA_TPCC_AETCMD	W	32	0x0000 0708	0x4151 0708
EDMA_TPCC_MPFAR	R	32	0x0000 0800	0x4151 0800
EDMA_TPCC_MPFSR	R	32	0x0000 0804	0x4151 0804
EDMA_TPCC_MPFCR	W	32	0x0000 0808	0x4151 0808
EDMA_TPCC_MPPAG	RW	32	0x0000 080C	0x4151 080C
EDMA_TPCC_MPPAN_k [6]	RW	32	0x0000 0810 + (0x4 * k)	0x4151 0810 + (0x4 * k)
EDMA_TPCC_ER	R	32	0x0000 1000	0x4151 1000
EDMA_TPCC_ERH	R	32	0x0000 1004	0x4151 1004
EDMA_TPCC_ECR	W	32	0x0000 1008	0x4151 1008
EDMA_TPCC_ECRH	W	32	0x0000 100C	0x4151 100C
EDMA_TPCC_ESR	W	32	0x0000 1010	0x4151 1010
EDMA_TPCC_ESRH	W	32	0x0000 1014	0x4151 1014
EDMA_TPCC_CER	R	32	0x0000 1018	0x4151 1018
EDMA_TPCC_CERH	R	32	0x0000 101C	0x4151 101C
EDMA_TPCC_EER	R	32	0x0000 1020	0x4151 1020
EDMA_TPCC_EERH	R	32	0x0000 1024	0x4151 1024
EDMA_TPCC_EECR	W	32	0x0000 1028	0x4151 1028
EDMA_TPCC_EECRH	W	32	0x0000 102C	0x4151 102C
EDMA_TPCC_EESR	W	32	0x0000 1030	0x4151 1030
EDMA_TPCC_EESRH	W	32	0x0000 1034	0x4151 1034
EDMA_TPCC_SER	R	32	0x0000 1038	0x4151 1038
EDMA_TPCC_SERH	R	32	0x0000 103C	0x4151 103C
EDMA_TPCC_SECR	W	32	0x0000 1040	0x4151 1040
EDMA_TPCC_SECRH	W	32	0x0000 1044	0x4151 1044
EDMA_TPCC_IER	R	32	0x0000 1050	0x4151 1050

(4) p = 0 to 15 for DSP2_EDMA_TPCC
(5) i = 0 to 1 for DSP2_EDMA_TPCC
(6) k = 0 to 7 for DSP2_EDMA_TPCC

Table 16-113. DSP2 EDMA_TPCC Registers Mapping Summary (L3_MAIN Access) (continued)

Register Name	Type	Register Width (Bits)	Address Offset	DSP2_EDMA_TPCC Physical Address (L3_MAIN Access)
EDMA_TPCC_IERH	R	32	0x0000 1054	0x4151 1054
EDMA_TPCC_IECR	W	32	0x0000 1058	0x4151 1058
EDMA_TPCC_IECRH	W	32	0x0000 105C	0x4151 105C
EDMA_TPCC_IESR	W	32	0x0000 1060	0x4151 1060
EDMA_TPCC_IESRH	W	32	0x0000 1064	0x4151 1064
EDMA_TPCC_IPR	R	32	0x0000 1068	0x4151 1068
EDMA_TPCC_IPRH	R	32	0x0000 106C	0x4151 106C
EDMA_TPCC_ICR	W	32	0x0000 1070	0x4151 1070
EDMA_TPCC_ICRH	W	32	0x0000 1074	0x4151 1074
EDMA_TPCC_IEVAL	W	32	0x0000 1078	0x4151 1078
EDMA_TPCC_QER	R	32	0x0000 1080	0x4151 1080
EDMA_TPCC_QEER	R	32	0x0000 1084	0x4151 1084
EDMA_TPCC_QEECR	W	32	0x0000 1088	0x4151 1088
EDMA_TPCC_QEESR	W	32	0x0000 108C	0x4151 108C
EDMA_TPCC_QSER	R	32	0x0000 1090	0x4151 1090
EDMA_TPCC_QSECR	W	32	0x0000 1094	0x4151 1094
EDMA_TPCC_ER_RN_k (6)	R	32	0x0000 2000 + (0x200 * k)	0x4151 2000 + (0x200 * k)
EDMA_TPCC_ERH_RN_k (6)	R	32	0x0000 2004 + (0x200 * k)	0x4151 2004 + (0x200 * k)
EDMA_TPCC_ECR_RN_k (6)	W	32	0x0000 2008 + (0x200 * k)	0x4151 2008 + (0x200 * k)
EDMA_TPCC_ECRH_RN_k (6)	W	32	0x0000 200C + (0x200 * k)	0x4151 200C + (0x200 * k)
EDMA_TPCC_ESR_RN_k (6)	W	32	0x0000 2010 + (0x200 * k)	0x4151 2010 + (0x200 * k)
EDMA_TPCC_ESRH_RN_k (6)	W	32	0x0000 2014 + (0x200 * k)	0x4151 2014 + (0x200 * k)
EDMA_TPCC_CER_RN_k (6)	R	32	0x0000 2018 + (0x200 * k)	0x4151 2018 + (0x200 * k)
EDMA_TPCC_CERH_RN_k (6)	R	32	0x0000 201C + (0x200 * k)	0x4151 201C + (0x200 * k)
EDMA_TPCC_EER_RN_k (6)	R	32	0x0000 2020 + (0x200 * k)	0x4151 2020 + (0x200 * k)
EDMA_TPCC_EERH_RN_k (6)	R	32	0x0000 2024 + (0x200 * k)	0x4151 2024 + (0x200 * k)
EDMA_TPCC_EECR_RN_k (7)	W	32	0x0000 2028 + (0x200 * k)	0x4151 2028 + (0x200 * k)
EDMA_TPCC_EECRH_RN_k (7)	W	32	0x0000 202C + (0x200 * k)	0x4151 202C + (0x200 * k)
EDMA_TPCC_EESR_RN_k (7)	W	32	0x0000 2030 + (0x200 * k)	0x4151 2030 + (0x200 * k)
EDMA_TPCC_EESRH_RN_k (7)	W	32	0x0000 2034 + (0x200 * k)	0x4151 2034 + (0x200 * k)
EDMA_TPCC_SER_RN_k (7)	R	32	0x0000 2038 + (0x200 * k)	0x4151 2038 + (0x200 * k)
EDMA_TPCC_SERH_RN_k (7)	R	32	0x0000 203C + (0x200 * k)	0x4151 203C + (0x200 * k)
EDMA_TPCC_SECR_RN_k (7)	W	32	0x0000 2040 + (0x200 * k)	0x4151 2040 + (0x200 * k)
EDMA_TPCC_SECRH_RN_k (7)	W	32	0x0000 2044 + (0x200 * k)	0x4151 2044 + (0x200 * k)
EDMA_TPCC_IER_RN_k (7)	R	32	0x0000 2050 + (0x200 * k)	0x4151 2050 + (0x200 * k)
EDMA_TPCC_IERH_RN_k (7)	R	32	0x0000 2054 + (0x200 * k)	0x4151 2054 + (0x200 * k)
EDMA_TPCC_IECR_RN_k (7)	W	32	0x0000 2058 + (0x200 * k)	0x4151 2058 + (0x200 * k)
EDMA_TPCC_IECRH_RN_k (7)	W	32	0x0000 205C + (0x200 * k)	0x4151 205C + (0x200 * k)
EDMA_TPCC_IESR_RN_k (7)	W	32	0x0000 2060 + (0x200 * k)	0x4151 2060 + (0x200 * k)
EDMA_TPCC_IESRH_RN_k (7)	W	32	0x0000 2064 + (0x200 * k)	0x4151 2064 + (0x200 * k)
EDMA_TPCC_IPR_RN_k (7)	R	32	0x0000 2068 + (0x200 * k)	0x4151 2068 + (0x200 * k)
EDMA_TPCC_IPRH_RN_k (7)	R	32	0x0000 206C + (0x200 * k)	0x4151 206C + (0x200 * k)
EDMA_TPCC_ICR_RN_k (7)	W	32	0x0000 2070 + (0x200 * k)	0x4151 2070 + (0x200 * k)
EDMA_TPCC_ICRH_RN_k (7)	W	32	0x0000 2074 + (0x200 * k)	0x4151 2074 + (0x200 * k)
EDMA_TPCC_IEVAL_RN_k (7)	W	32	0x0000 2078 + (0x200 * k)	0x4151 2078 + (0x200 * k)

(7) k = 0 to 7 for DSP2_EDMA_TPCC

Table 16-113. DSP2 EDMA_TPCC Registers Mapping Summary (L3_MAIN Access) (continued)

Register Name	Type	Register Width (Bits)	Address Offset	DSP2_EDMA_TPCC Physical Address (L3_MAIN Access)
EDMA_TPCC_QER_RN_k [7]	R	32	0x0000 2080 + (0x200 * k)	0x4151 2080 + (0x200 * k)
EDMA_TPCC_QEER_RN_k [7]	R	32	0x0000 2084 + (0x200 * k)	0x4151 2084 + (0x200 * k)
EDMA_TPCC_QEECR_RN_k [7]	W	32	0x0000 2088 + (0x200 * k)	0x4151 2088 + (0x200 * k)
EDMA_TPCC_QEESR_RN_k [7]	W	32	0x0000 208C + (0x200 * k)	0x4151 208C + (0x200 * k)
EDMA_TPCC_QSER_RN_k [7]	R	32	0x0000 2090 + (0x200 * k)	0x4151 2090 + (0x200 * k)
EDMA_TPCC_QSECR_RN_k [7]	W	32	0x0000 2094 + (0x200 * k)	0x4151 2094 + (0x200 * k)
EDMA_TPCC_OPT_n [8]	RW	32	0x0000 4000 + (0x20 * n)	0x4151 4000 + (0x20 * n)
EDMA_TPCC_SRC_n [8]	RW	32	0x0000 4004 + (0x20 * n)	0x4151 4004 + (0x20 * n)
EDMA_TPCC_ABCNT_n [8]	RW	32	0x0000 4008 + (0x20 * n)	0x4151 4008 + (0x20 * n)
EDMA_TPCC_DST_n [8]	RW	32	0x0000 400C + (0x20 * n)	0x4151 400C + (0x20 * n)
EDMA_TPCC_BIDX_n [8]	RW	32	0x0000 4010 + (0x20 * n)	0x4151 4010 + (0x20 * n)
EDMA_TPCC_LNK_n [8]	RW	32	0x0000 4014 + (0x20 * n)	0x4151 4014 + (0x20 * n)
EDMA_TPCC_CIDX_n [8]	RW	32	0x0000 4018 + (0x20 * n)	0x4151 4018 + (0x20 * n)
EDMA_TPCC_CCNT_n [8]	RW	32	0x0000 401C + (0x20 * n)	0x4151 401C + (0x20 * n)

(8) n = 0 to 127 for DSP2_EDMA_TPCC

Table 16-114. DSP EDMA_TPCC Registers Mapping Summary (Private Access)

Register Name	Type	Register Width (Bits)	Address Offset	DSP_EDMA_TPCC Physical Address (DSP Private Access)
EDMA_TPCC_PID	R	32	0x0000 0000	0x01D1 0000
EDMA_TPCC_CCCFG	R	32	0x0000 0004	0x01D1 0004
EDMA_TPCC_CLKGDIS	RW	32	0x0000 00FC	0x01D1 00FC
EDMA_TPCC_DCHMAPN_m [1]	RW	32	0x0000 0100 + (0x4 * m)	0x01D1 0100 + (0x4 * m)
EDMA_TPCC_QCHMAPN_j [2]	RW	32	0x0000 0200 + (0x4 * j)	0x01D1 0200 + (0x4 * j)
EDMA_TPCC_DMAQNUMN_k [3]	RW	32	0x0000 0240 + (0x4 * k)	0x01D1 0240 + (0x4 * k)
EDMA_TPCC_QDMAQNUM	RW	32	0x0000 0260	0x01D1 0260
EDMA_TPCC_QUETCMAP	RW	32	0x0000 0280	0x01D1 0280
EDMA_TPCC_QUEPRI	RW	32	0x0000 0284	0x01D1 0284
EDMA_TPCC_EMR	R	32	0x0000 0300	0x01D1 0300
EDMA_TPCC_EMRH	R	32	0x0000 0304	0x01D1 0304
EDMA_TPCC_EMCR	W	32	0x0000 0308	0x01D1 0308
EDMA_TPCC_EMCRH	W	32	0x0000 030C	0x01D1 030C
EDMA_TPCC_QEMR	R	32	0x0000 0310	0x01D1 0310
EDMA_TPCC_QEMCR	W	32	0x0000 0314	0x01D1 0314
EDMA_TPCC_CCERR	R	32	0x0000 0318	0x01D1 0318
EDMA_TPCC_CCERRCLR	W	32	0x0000 031C	0x01D1 031C
EDMA_TPCC_EEVAL	W	32	0x0000 0320	0x01D1 0320
EDMA_TPCC_DRAEM_k [3]	RW	32	0x0000 0340 + (0x8 * k)	0x01D1 0340 + (0x8 * k)
EDMA_TPCC_DRAEHM_k [3]	RW	32	0x0000 0344 + (0x8 * k)	0x01D1 0344 + (0x8 * k)
EDMA_TPCC_QRAEN_k [3]	RW	32	0x0000 0380 + (0x4 * k)	0x01D1 0380 + (0x4 * k)
EDMA_TPCC_Q0E_p [4]	R	32	0x0000 0400 + (0x4 *pl)	0x01D1 0400 + (0x4 * p)
EDMA_TPCC_Q1E_p [4]	R	32	0x0000 0440 + (0x4 * p)	0x01D1 0440 + (0x4 * p)

(1) m = 0 to 63 for DSP_EDMA_TPCC
(2) j = 0 to 7 for DSP_EDMA_TPCC
(3) k = 0 to 7 for DSP_EDMA_TPCC
(4) p = 0 to 15 for DSP_EDMA_TPCC

Table 16-114. DSP EDMA_TPCC Registers Mapping Summary (Private Access) (continued)

Register Name	Type	Register Width (Bits)	Address Offset	DSP_EDMA_TPCC Physical Address (DSP Private Access)
EDMA_TPCC_QSTATN_i [(5)]	R	32	0x0000 0600 + (0x4 * i)	0x01D1 0600 + (0x4 * i)
EDMA_TPCC_QWMTHRA	RW	32	0x0000 0620	0x01D1 0620
EDMA_TPCC_QWMTHRB	RW	32	0x0000 0624	0x01D1 0624
EDMA_TPCC_CCSTAT	R	32	0x0000 0640	0x01D1 0640
EDMA_TPCC_AETCTL	RW	32	0x0000 0700	0x01D1 0700
EDMA_TPCC_AETSTAT	R	32	0x0000 0704	0x01D1 0704
EDMA_TPCC_AETCMD	W	32	0x0000 0708	0x01D1 0708
EDMA_TPCC_MPFAR	R	32	0x0000 0800	0x01D1 0800
EDMA_TPCC_MPFSR	R	32	0x0000 0804	0x01D1 0804
EDMA_TPCC_MPFCR	W	32	0x0000 0808	0x01D1 0808
EDMA_TPCC_MPPAG	RW	32	0x0000 080C	0x01D1 080C
EDMA_TPCC_MPPAN_k [(3)]	RW	32	0x0000 0810 + (0x4 * k)	0x01D1 0810 + (0x4 * k)
EDMA_TPCC_ER	R	32	0x0000 1000	0x01D1 1000
EDMA_TPCC_ERH	R	32	0x0000 1004	0x01D1 1004
EDMA_TPCC_ECR	W	32	0x0000 1008	0x01D1 1008
EDMA_TPCC_ECRH	W	32	0x0000 100C	0x01D1 100C
EDMA_TPCC_ESR	W	32	0x0000 1010	0x01D1 1010
EDMA_TPCC_ESRH	W	32	0x0000 1014	0x01D1 1014
EDMA_TPCC_CER	R	32	0x0000 1018	0x01D1 1018
EDMA_TPCC_CERH	R	32	0x0000 101C	0x01D1 101C
EDMA_TPCC_EER	R	32	0x0000 1020	0x01D1 1020
EDMA_TPCC_EERH	R	32	0x0000 1024	0x01D1 1024
EDMA_TPCC_EECR	W	32	0x0000 1028	0x01D1 1028
EDMA_TPCC_EECRH	W	32	0x0000 102C	0x01D1 102C
EDMA_TPCC_EESR	W	32	0x0000 1030	0x01D1 1030
EDMA_TPCC_EESRH	W	32	0x0000 1034	0x01D1 1034
EDMA_TPCC_SER	R	32	0x0000 1038	0x01D1 1038
EDMA_TPCC_SERH	R	32	0x0000 103C	0x01D1 103C
EDMA_TPCC_SECR	W	32	0x0000 1040	0x01D1 1040
EDMA_TPCC_SECRH	W	32	0x0000 1044	0x01D1 1044
EDMA_TPCC_IER	R	32	0x0000 1050	0x01D1 1050
EDMA_TPCC_IERH	R	32	0x0000 1054	0x01D1 1054
EDMA_TPCC_IECR	W	32	0x0000 1058	0x01D1 1058
EDMA_TPCC_IECRH	W	32	0x0000 105C	0x01D1 105C
EDMA_TPCC_IESR	W	32	0x0000 1060	0x01D1 1060
EDMA_TPCC_IESRH	W	32	0x0000 1064	0x01D1 1064
EDMA_TPCC_IPR	R	32	0x0000 1068	0x01D1 1068
EDMA_TPCC_IPRH	R	32	0x0000 106C	0x01D1 106C
EDMA_TPCC_ICR	W	32	0x0000 1070	0x01D1 1070
EDMA_TPCC_ICRH	W	32	0x0000 1074	0x01D1 1074
EDMA_TPCC_IEVAL	W	32	0x0000 1078	0x01D1 1078
EDMA_TPCC_QER	R	32	0x0000 1080	0x01D1 1080
EDMA_TPCC_QEER	R	32	0x0000 1084	0x01D1 1084
EDMA_TPCC_QEECR	W	32	0x0000 1088	0x01D1 1088
EDMA_TPCC_QEESR	W	32	0x0000 108C	0x01D1 108C

(5) i = 0 to 1 for DSP_EDMA_TPCC

Table 16-114. DSP EDMA_TPCC Registers Mapping Summary (Private Access) (continued)

Register Name	Type	Register Width (Bits)	Address Offset	DSP_EDMA_TPCC Physical Address (DSP Private Access)
EDMA_TPCC_QSER	R	32	0x0000 1090	0x01D1 1090
EDMA_TPCC_QSECR	W	32	0x0000 1094	0x01D1 1094
EDMA_TPCC_ER_RN_k [6]	R	32	0x0000 2000 + (0x200 * k)	0x01D1 2000 + (0x200 * k)
EDMA_TPCC_ERH_RN_k [6]	R	32	0x0000 2004 + (0x200 * k)	0x01D1 2004 + (0x200 * k)
EDMA_TPCC_ECR_RN_k [6]	W	32	0x0000 2008 + (0x200 * k)	0x01D1 2008 + (0x200 * k)
EDMA_TPCC_ECRH_RN_k [6]	W	32	0x0000 200C + (0x200 * k)	0x01D1 200C + (0x200 * k)
EDMA_TPCC_ESR_RN_k [6]	W	32	0x0000 2010 + (0x200 * k)	0x01D1 2010 + (0x200 * k)
EDMA_TPCC_ESRH_RN_k [6]	W	32	0x0000 2014 + (0x200 * k)	0x01D1 2014 + (0x200 * k)
EDMA_TPCC_CER_RN_k [6]	R	32	0x0000 2018 + (0x200 * k)	0x01D1 2018 + (0x200 * k)
EDMA_TPCC_CERH_RN_k [6]	R	32	0x0000 201C + (0x200 * k)	0x01D1 201C + (0x200 * k)
EDMA_TPCC_EER_RN_k [6]	R	32	0x0000 2020 + (0x200 * k)	0x01D1 2020 + (0x200 * k)
EDMA_TPCC_EERH_RN_k [6]	R	32	0x0000 2024 + (0x200 * k)	0x01D1 2024 + (0x200 * k)
EDMA_TPCC_EECR_RN_k [6]	W	32	0x0000 2028 + (0x200 * k)	0x01D1 2028 + (0x200 * k)
EDMA_TPCC_EECRH_RN_k [6]	W	32	0x0000 202C + (0x200 * k)	0x01D1 202C + (0x200 * k)
EDMA_TPCC_EESR_RN_k [6]	W	32	0x0000 2030 + (0x200 * k)	0x01D1 2030 + (0x200 * k)
EDMA_TPCC_EESRH_RN_k [6]	W	32	0x0000 2034 + (0x200 * k)	0x01D1 2034 + (0x200 * k)
EDMA_TPCC_SER_RN_k [6]	R	32	0x0000 2038 + (0x200 * k)	0x01D1 2038 + (0x200 * k)
EDMA_TPCC_SERH_RN_k [6]	R	32	0x0000 203C + (0x200 * k)	0x01D1 203C + (0x200 * k)
EDMA_TPCC_SECR_RN_k [6]	W	32	0x0000 2040 + (0x200 * k)	0x01D1 2040 + (0x200 * k)
EDMA_TPCC_SECRH_RN_k [6]	W	32	0x0000 2044 + (0x200 * k)	0x01D1 2044 + (0x200 * k)
EDMA_TPCC_IER_RN_k [6]	R	32	0x0000 2050 + (0x200 * k)	0x01D1 2050 + (0x200 * k)
EDMA_TPCC_IERH_RN_k [6]	R	32	0x0000 2054 + (0x200 * k)	0x01D1 2054 + (0x200 * k)
EDMA_TPCC_IECR_RN_k [6]	W	32	0x0000 2058 + (0x200 * k)	0x01D1 2058 + (0x200 * k)
EDMA_TPCC_IECRH_RN_k [6]	W	32	0x0000 205C + (0x200 * k)	0x01D1 205C + (0x200 * k)
EDMA_TPCC_IESR_RN_k [6]	W	32	0x0000 2060 + (0x200 * k)	0x01D1 2060 + (0x200 * k)
EDMA_TPCC_IESRH_RN_k [7]	W	32	0x0000 2064 + (0x200 * k)	0x01D1 2064 + (0x200 * k)
EDMA_TPCC_IPR_RN_k [7]	R	32	0x0000 2068 + (0x200 * k)	0x01D1 2068 + (0x200 * k)
EDMA_TPCC_IPRH_RN_k [7]	R	32	0x0000 206C + (0x200 * k)	0x01D1 206C + (0x200 * k)
EDMA_TPCC_ICR_RN_k [7]	W	32	0x0000 2070 + (0x200 * k)	0x01D1 2070 + (0x200 * k)
EDMA_TPCC_ICRH_RN_k [7]	W	32	0x0000 2074 + (0x200 * k)	0x01D1 2074 + (0x200 * k)
EDMA_TPCC_IEVAL_RN_k [7]	W	32	0x0000 2078 + (0x200 * k)	0x01D1 2078 + (0x200 * k)
EDMA_TPCC_QER_RN_k [7]	R	32	0x0000 2080 + (0x200 * k)	0x01D1 2080 + (0x200 * k)
EDMA_TPCC_QEER_RN_k [7]	R	32	0x0000 2084 + (0x200 * k)	0x01D1 2084 + (0x200 * k)
EDMA_TPCC_QEECR_RN_k [7]	W	32	0x0000 2088 + (0x200 * k)	0x01D1 2088 + (0x200 * k)
EDMA_TPCC_QEESR_RN_k [7]	W	32	0x0000 208C + (0x200 * k)	0x01D1 208C + (0x200 * k)
EDMA_TPCC_QSER_RN_k [7]	R	32	0x0000 2090 + (0x200 * k)	0x01D1 2090 + (0x200 * k)
EDMA_TPCC_QSECR_RN_k [7]	W	32	0x0000 2094 + (0x200 * k)	0x01D1 2094 + (0x200 * k)
EDMA_TPCC_OPT_n [8]	RW	32	0x0000 4000 + (0x20 * n)	0x01D1 4000 + (0x20 * n)
EDMA_TPCC_SRC_n [8]	RW	32	0x0000 4004 + (0x20 * n)	0x01D1 4004 + (0x20 * n)
EDMA_TPCC_ABCNT_n [8]	RW	32	0x0000 4008 + (0x20 * n)	0x01D1 4008 + (0x20 * n)
EDMA_TPCC_DST_n [8]	RW	32	0x0000 400C + (0x20 * n)	0x01D1 400C + (0x20 * n)
EDMA_TPCC_BIDX_n [8]	RW	32	0x0000 4010 + (0x20 * n)	0x01D1 4010 + (0x20 * n)
EDMA_TPCC_LNK_n [8]	RW	32	0x0000 4014 + (0x20 * n)	0x01D1 4014 + (0x20 * n)
EDMA_TPCC_CIDX_n [8]	RW	32	0x0000 4018 + (0x20 * n)	0x01D1 4018 + (0x20 * n)

(6) k = 0 to 7 for DSP_EDMA_TPCC
(7) k = 0 to 7 for DSP_EDMA_TPCC
(8) n = 0 to 127 for DSP_EDMA_TPCC

Table 16-114. DSP EDMA_TPCC Registers Mapping Summary (Private Access) (continued)

Register Name	Type	Register Width (Bits)	Address Offset	DSP_EDMA_TPCC Physical Address (DSP Private Access)
EDMA_TPCC_CCNT_n [8]	RW	32	0x0000 401C + (0x20 * n)	0x01D1 401C + (0x20 * n)

NOTE: EVE is not supported in this family of devices.

Table 16-115. EVE1 and EVE2 EDMA_TPCC Registers Mapping Summary (L3_MAIN Access)

Register Name	Type	Register Width (Bits)	Address Offset	EVE1_EDMA_TPCC Physical Address (L3_MAIN Access)	EVE2_EDMA_TPCC Physical Address (L3_MAIN Access)
EDMA_TPCC_PID	R	32	0x0000 0000	0x420A 0000	0x421A 0000
EDMA_TPCC_CCCFG	R	32	0x0000 0004	0x420A 0004	0x421A 0004
EDMA_TPCC_CLKGDIS	RW	32	0x0000 00FC	0x420A 00FC	0x421A 00FC
EDMA_TPCC_DCHMAPN_m [1]	RW	32	0x0000 0100 + (0x4 * m)	0x420A 0100 + (0x4 * m)	0x421A 0100 + (0x4 * m)
EDMA_TPCC_QCHMAPN_j [2]	RW	32	0x0000 0200 + (0x4 * j)	0x420A 0200 + (0x4 * j)	0x421A 0200 + (0x4 * j)
EDMA_TPCC_DMAQNUMN_k [3]	RW	32	0x0000 0240 + (0x4 * k)	0x420A 0240 + (0x4 * k)	0x421A 0240 + (0x4 * k)
EDMA_TPCC_QDMAQNUM	RW	32	0x0000 0260	0x420A 0260	0x421A 0260
EDMA_TPCC_QUETCMAP	RW	32	0x0000 0280	0x420A 0280	0x421A 0280
EDMA_TPCC_QUEPRI	RW	32	0x0000 0284	0x420A 0284	0x421A 0284
EDMA_TPCC_EMR	R	32	0x0000 0300	0x420A 0300	0x421A 0300
EDMA_TPCC_EMRH	R	32	0x0000 0304	0x420A 0304	0x421A 0304
EDMA_TPCC_EMCR	W	32	0x0000 0308	0x420A 0308	0x421A 0308
EDMA_TPCC_EMCRH	W	32	0x0000 030C	0x420A 030C	0x421A 030C
EDMA_TPCC_QEMR	R	32	0x0000 0310	0x420A 0310	0x421A 0310
EDMA_TPCC_QEMCR	W	32	0x0000 0314	0x420A 0314	0x421A 0314
EDMA_TPCC_CCERR	R	32	0x0000 0318	0x420A 0318	0x421A 0318
EDMA_TPCC_CCERRCLR	W	32	0x0000 031C	0x420A 031C	0x421A 031C
EDMA_TPCC_EEVAL	W	32	0x0000 0320	0x420A 0320	0x421A 0320
EDMA_TPCC_DRAEM_k [3]	RW	32	0x0000 0340 + (0x8 * k)	-	-
EDMA_TPCC_DRAEHM_k [3]	RW	32	0x0000 0344 + (0x8 * k)	-	-
EDMA_TPCC_QRAEN_k [3]	RW	32	0x0000 0380 + (0x4 * k)	0x420A 0380 + (0x4 * k)	0x421A 0380 + (0x4 * k)
EDMA_TPCC_Q0E_p [4]	R	32	0x0000 0400 + (0x4 * p)	-	-
EDMA_TPCC_Q1E_p [4]	R	32	0x0000 0440 + (0x4 *pl)	-	-
EDMA_TPCC_QSTATN_i [5]	R	32	0x0000 0600 + (0x4 * i)	0x420A 0600 + (0x4 * i)	0x421A 0600 + (0x4 * i)
EDMA_TPCC_QWMTHRA	RW	32	0x0000 0620	0x420A 0620	0x421A 0620
EDMA_TPCC_QWMTHRB	RW	32	0x0000 0624	0x420A 0624	0x421A 0624
EDMA_TPCC_CCSTAT	R	32	0x0000 0640	0x420A 0640	0x421A 0640
EDMA_TPCC_AETCTL	RW	32	0x0000 0700	0x420A 0700	0x421A 0700
EDMA_TPCC_AETSTAT	R	32	0x0000 0704	0x420A 0704	0x421A 0704
EDMA_TPCC_AETCMD	W	32	0x0000 0708	0x420A 0708	0x421A 0708
EDMA_TPCC_MPFAR	R	32	0x0000 0800	0x420A 0800	0x421A 0800
EDMA_TPCC_MPFSR	R	32	0x0000 0804	0x420A 0804	0x421A 0804

(1) m = 0 to 15 for EVE1_EDMA_TPCC and EVE2_EDMA_TPCC
(2) j = 0 to 7 for EVE1_EDMA_TPCC and EVE2_EDMA_TPCC
(3) k = 0 to 7 for EVE1_EDMA_TPCC and EVE2_EDMA_TPCC
(4) p = 0 to 1 for EVE1_EDMA_TPCC and EVE2_EDMA_TPCC
(5) i = 0 to 1 for EVE1_EDMA_TPCC and EVE2_EDMA_TPCC

Table 16-115. EVE1 and EVE2 EDMA_TPCC Registers Mapping Summary (L3_MAIN Access) (continued)

Register Name	Type	Register Width (Bits)	Address Offset	EVE1_EDMA_TPCC Physical Address (L3_MAIN Access)	EVE2_EDMA_TPCC Physical Address (L3_MAIN Access)
EDMA_TPCC_MPFCR	W	32	0x0000 0808	0x420A 0808	0x421A 0808
EDMA_TPCC_MPPAG	RW	32	0x0000 080C	0x420A 080C	0x421A 080C
EDMA_TPCC_MPPAN_k [(3)]	RW	32	0x0000 0810 + (0x4 * k)	0x420A 0810 + (0x4 * k)	0x421A 0810 + (0x4 * k)
EDMA_TPCC_ER	R	32	0x0000 1000	0x420A 1000	0x421A 1000
EDMA_TPCC_ERH	R	32	0x0000 1004	0x420A 1004	0x421A 1004
EDMA_TPCC_ECR	W	32	0x0000 1008	0x420A 1008	0x421A 1008
EDMA_TPCC_ECRH	W	32	0x0000 100C	0x420A 100C	0x421A 100C
EDMA_TPCC_ESR	W	32	0x0000 1010	0x420A 1010	0x421A 1010
EDMA_TPCC_ESRH	W	32	0x0000 1014	0x420A 1014	0x421A 1014
EDMA_TPCC_CER	R	32	0x0000 1018	0x420A 1018	0x421A 1018
EDMA_TPCC_CERH	R	32	0x0000 101C	0x420A 101C	0x421A 101C
EDMA_TPCC_EER	R	32	0x0000 1020	0x420A 1020	0x421A 1020
EDMA_TPCC_EERH	R	32	0x0000 1024	0x420A 1024	0x421A 1024
EDMA_TPCC_EECR	W	32	0x0000 1028	0x420A 1028	0x421A 1028
EDMA_TPCC_EECRH	W	32	0x0000 102C	0x420A 102C	0x421A 102C
EDMA_TPCC_EESR	W	32	0x0000 1030	0x420A 1030	0x421A 1030
EDMA_TPCC_EESRH	W	32	0x0000 1034	0x420A 1034	0x421A 1034
EDMA_TPCC_SER	R	32	0x0000 1038	0x420A 1038	0x421A 1038
EDMA_TPCC_SERH	R	32	0x0000 103C	0x420A 103C	0x421A 103C
EDMA_TPCC_SECR	W	32	0x0000 1040	0x420A 1040	0x421A 1040
EDMA_TPCC_SECRH	W	32	0x0000 1044	0x420A 1044	0x421A 1044
EDMA_TPCC_IER	R	32	0x0000 1050	0x420A 1050	0x421A 1050
EDMA_TPCC_IERH	R	32	0x0000 1054	0x420A 1054	0x421A 1054
EDMA_TPCC_IECR	W	32	0x0000 1058	0x420A 1058	0x421A 1058
EDMA_TPCC_IECRH	W	32	0x0000 105C	0x420A 105C	0x421A 105C
EDMA_TPCC_IESR	W	32	0x0000 1060	0x420A 1060	0x421A 1060
EDMA_TPCC_IESRH	W	32	0x0000 1064	0x420A 1064	0x421A 1064
EDMA_TPCC_IPR	R	32	0x0000 1068	0x420A 1068	0x421A 1068
EDMA_TPCC_IPRH	R	32	0x0000 106C	0x420A 106C	0x421A 106C
EDMA_TPCC_ICR	W	32	0x0000 1070	0x420A 1070	0x421A 1070
EDMA_TPCC_ICRH	W	32	0x0000 1074	0x420A 1074	0x421A 1074
EDMA_TPCC_IEVAL	W	32	0x0000 1078	0x420A 1078	0x421A 1078
EDMA_TPCC_QER	R	32	0x0000 1080	0x420A 1080	0x421A 1080
EDMA_TPCC_QEER	R	32	0x0000 1084	0x420A 1084	0x421A 1084
EDMA_TPCC_QEECR	W	32	0x0000 1088	0x420A 1088	0x421A 1088
EDMA_TPCC_QEESR	W	32	0x0000 108C	0x420A 108C	0x421A 108C
EDMA_TPCC_QSER	R	32	0x0000 1090	0x420A 1090	0x421A 1090
EDMA_TPCC_QSECR	W	32	0x0000 1094	0x420A 1094	0x421A 1094
EDMA_TPCC_ER_RN_k [(6)]	R	32	0x0000 2000 + (0x200 * k)	0x420A 2000 + (0x200 * k)	0x421A 2000 + (0x200 * k)
EDMA_TPCC_ERH_RN_k [(6)]	R	32	0x0000 2004 + (0x200 * k)	0x420A 2004 + (0x200 * k)	0x421A 2004 + (0x200 * k)
EDMA_TPCC_ECR_RN_k [(6)]	W	32	0x0000 2008 + (0x200 * k)	0x420A 2008 + (0x200 * k)	0x421A 2008 + (0x200 * k)
EDMA_TPCC_ECRH_RN_k [(6)]	W	32	0x0000 200C + (0x200 * k)	0x420A 200C + (0x200 * k)	0x421A 200C + (0x200 * k)

(6) k = 0 to 7 for EVE1_EDMA_TPCC and EVE2_EDMA_TPCC

Table 16-115. EVE1 and EVE2 EDMA_TPCC Registers Mapping Summary (L3_MAIN Access) (continued)

Register Name	Type	Register Width (Bits)	Address Offset	EVE1_EDMA_TPCC Physical Address (L3_MAIN Access)	EVE2_EDMA_TPCC Physical Address (L3_MAIN Access)
EDMA_TPCC_ESR_RN_k (6)	W	32	0x0000 2010 + (0x200 * k)	0x420A 2010 + (0x200 * k)	0x421A 2010 + (0x200 * k)
EDMA_TPCC_ESRH_RN_k (6)	W	32	0x0000 2014 + (0x200 * k)	0x420A 2014 + (0x200 * k)	0x421A 2014 + (0x200 * k)
EDMA_TPCC_CER_RN_k (6)	R	32	0x0000 2018 + (0x200 * k)	0x420A 2018 + (0x200 * k)	0x421A 2018 + (0x200 * k)
EDMA_TPCC_CERH_RN_k (6)	R	32	0x0000 201C + (0x200 * k)	0x420A 201C + (0x200 * k)	0x421A 201C + (0x200 * k)
EDMA_TPCC_EER_RN_k (6)	R	32	0x0000 2020 + (0x200 * k)	0x420A 2020 + (0x200 * k)	0x421A 2020 + (0x200 * k)
EDMA_TPCC_EERH_RN_k (6)	R	32	0x0000 2024 + (0x200 * k)	0x420A 2024 + (0x200 * k)	0x421A 2024 + (0x200 * k)
EDMA_TPCC_EECR_RN_k (6)	W	32	0x0000 2028 + (0x200 * k)	0x420A 2028 + (0x200 * k)	0x421A 2028 + (0x200 * k)
EDMA_TPCC_EECRH_RN_k (6)	W	32	0x0000 202C + (0x200 * k)	0x420A 202C + (0x200 * k)	0x421A 202C + (0x200 * k)
EDMA_TPCC_EESR_RN_k (6)	W	32	0x0000 2030 + (0x200 * k)	0x420A 2030 + (0x200 * k)	0x421A 2030 + (0x200 * k)
EDMA_TPCC_EESRH_RN_k (6)	W	32	0x0000 2034 + (0x200 * k)	0x420A 2034 + (0x200 * k)	0x421A 2034 + (0x200 * k)
EDMA_TPCC_SER_RN_k (6)	R	32	0x0000 2038 + (0x200 * k)	0x420A 2038 + (0x200 * k)	0x421A 2038 + (0x200 * k)
EDMA_TPCC_SERH_RN_k (6)	R	32	0x0000 203C + (0x200 * k)	0x420A 203C + (0x200 * k)	0x421A 203C + (0x200 * k)
EDMA_TPCC_SECR_RN_k (6)	W	32	0x0000 2040 + (0x200 * k)	0x420A 2040 + (0x200 * k)	0x421A 2040 + (0x200 * k)
EDMA_TPCC_SECRH_RN_k (6)	W	32	0x0000 2044 + (0x200 * k)	0x420A 2044 + (0x200 * k)	0x421A 2044 + (0x200 * k)
EDMA_TPCC_IER_RN_k (6)	R	32	0x0000 2050 + (0x200 * k)	0x420A 2050 + (0x200 * k)	0x421A 2050 + (0x200 * k)
EDMA_TPCC_IERH_RN_k (7)	R	32	0x0000 2054 + (0x200 * k)	0x420A 2054 + (0x200 * k)	0x421A 2054 + (0x200 * k)
EDMA_TPCC_IECR_RN_k (7)	W	32	0x0000 2058 + (0x200 * k)	0x420A 2058 + (0x200 * k)	0x421A 2058 + (0x200 * k)
EDMA_TPCC_IECRH_RN_k (7)	W	32	0x0000 205C + (0x200 * k)	0x420A 205C + (0x200 * k)	0x421A 205C + (0x200 * k)
EDMA_TPCC_IESR_RN_k (7)	W	32	0x0000 2060 + (0x200 * k)	0x420A 2060 + (0x200 * k)	0x421A 2060 + (0x200 * k)
EDMA_TPCC_IESRH_RN_k (7)	W	32	0x0000 2064 + (0x200 * k)	0x420A 2064 + (0x200 * k)	0x421A 2064 + (0x200 * k)
EDMA_TPCC_IPR_RN_k (7)	R	32	0x0000 2068 + (0x200 * k)	0x420A 2068 + (0x200 * k)	0x421A 2068 + (0x200 * k)
EDMA_TPCC_IPRH_RN_k (7)	R	32	0x0000 206C + (0x200 * k)	0x420A 206C + (0x200 * k)	0x421A 206C + (0x200 * k)
EDMA_TPCC_ICR_RN_k (7)	W	32	0x0000 2070 + (0x200 * k)	0x420A 2070 + (0x200 * k)	0x421A 2070 + (0x200 * k)
EDMA_TPCC_ICRH_RN_k (7)	W	32	0x0000 2074 + (0x200 * k)	0x420A 2074 + (0x200 * k)	0x421A 2074 + (0x200 * k)
EDMA_TPCC_IEVAL_RN_k (7)	W	32	0x0000 2078 + (0x200 * k)	0x420A 2078 + (0x200 * k)	0x421A 2078 + (0x200 * k)
EDMA_TPCC_QER_RN_k (7)	R	32	0x0000 2080 + (0x200 * k)	0x420A 2080 + (0x200 * k)	0x421A 2080 + (0x200 * k)

(7) k = 0 to 7 for EVE1_EDMA_TPCC and EVE2_EDMA_TPCC

Table 16-115. EVE1 and EVE2 EDMA_TPCC Registers Mapping Summary (L3_MAIN Access) (continued)

Register Name	Type	Register Width (Bits)	Address Offset	EVE1_EDMA_TPCC Physical Address (L3_MAIN Access)	EVE2_EDMA_TPCC Physical Address (L3_MAIN Access)
EDMA_TPCC_QEER_RN_k [7]	R	32	0x0000 2084 + (0x200 * k)	0x420A 2084 + (0x200 * k)	0x421A 2084 + (0x200 * k)
EDMA_TPCC_QEECR_RN_k [7]	W	32	0x0000 2088 + (0x200 * k)	0x420A 2088 + (0x200 * k)	0x421A 2088 + (0x200 * k)
EDMA_TPCC_QEESR_RN_k [7]	W	32	0x0000 208C + (0x200 * k)	0x420A 208C + (0x200 * k)	0x421A 208C + (0x200 * k)
EDMA_TPCC_QSER_RN_k [7]	R	32	0x0000 2090 + (0x200 * k)	0x420A 2090 + (0x200 * k)	0x421A 2090 + (0x200 * k)
EDMA_TPCC_QSECR_RN_k [7]	W	32	0x0000 2094 + (0x200 * k)	0x420A 2094 + (0x200 * k)	0x421A 2094 + (0x200 * k)
EDMA_TPCC_OPT_n [8]	RW	32	0x0000 4000 + (0x20 * n)	0x420A 4000 + (0x20 * n)	0x421A 4000 + (0x20 * n)
EDMA_TPCC_SRC_n [8]	RW	32	0x0000 4004 + (0x20 * n)	0x420A 4004 + (0x20 * n)	0x421A 4004 + (0x20 * n)
EDMA_TPCC_ABCNT_n [8]	RW	32	0x0000 4008 + (0x20 * n)	0x420A 4008 + (0x20 * n)	0x421A 4008 + (0x20 * n)
EDMA_TPCC_DST_n [8]	RW	32	0x0000 400C + (0x20 * n)	0x420A 400C + (0x20 * n)	0x421A 400C + (0x20 * n)
EDMA_TPCC_BIDX_n [8]	RW	32	0x0000 4010 + (0x20 * n)	0x420A 4010 + (0x20 * n)	0x421A 4010 + (0x20 * n)
EDMA_TPCC_LNK_n [8]	RW	32	0x0000 4014 + (0x20 * n)	0x420A 4014 + (0x20 * n)	0x421A 4014 + (0x20 * n)
EDMA_TPCC_CIDX_n [8]	RW	32	0x0000 4018 + (0x20 * n)	0x420A 4018 + (0x20 * n)	0x421A 4018 + (0x20 * n)
EDMA_TPCC_CCNT_n [8]	RW	32	0x0000 401C + (0x20 * n)	0x420A 401C + (0x20 * n)	0x421A 401C + (0x20 * n)

(8) n = 0 to 127 for EVE1_EDMA_TPCC and EVE2_EDMA_TPCC

NOTE: EVE is not supported in this family of devices.

Table 16-116. EVE3 and EVE4 EDMA_TPCC Registers Mapping Summary (L3_MAIN Access)

Register Name	Type	Register Width (Bits)	Address Offset	EVE3_EDMA_TPCC Physical Address (L3_MAIN Access)	EVE4_EDMA_TPCC Physical Address (L3_MAIN Access)
EDMA_TPCC_PID	R	32	0x0000 0000	0x422A 0000	0x423A 0000
EDMA_TPCC_CCCFG	R	32	0x0000 0004	0x422A 0004	0x423A 0004
EDMA_TPCC_CLKGDIS	RW	32	0x0000 00FC	0x422A 00FC	0x423A 00FC
EDMA_TPCC_DCHMAPN_m [1]	RW	32	0x0000 0100 + (0x4 * m)	0x422A 0100 + (0x4 * m)	0x423A 0100 + (0x4 * m)
EDMA_TPCC_QCHMAPN_j [2]	RW	32	0x0000 0200 + (0x4 * j)	0x422A 0200 + (0x4 * j)	0x423A 0200 + (0x4 * j)
EDMA_TPCC_DMAQNUMN_k [3]	RW	32	0x0000 0240 + (0x4 * k)	0x422A 0240 + (0x4 * k)	0x423A 0240 + (0x4 * k)
EDMA_TPCC_QDMAQNUM	RW	32	0x0000 0260	0x422A 0260	0x423A 0260
EDMA_TPCC_QUETCMAP	RW	32	0x0000 0280	0x422A 0280	0x423A 0280
EDMA_TPCC_QUEPRI	RW	32	0x0000 0284	0x422A 0284	0x423A 0284
EDMA_TPCC_EMR	R	32	0x0000 0300	0x422A 0300	0x423A 0300
EDMA_TPCC_EMRH	R	32	0x0000 0304	0x422A 0304	0x423A 0304

(1) m = 0 to 63 for EVE3_EDMA_TPCC and EVE4_EDMA_TPCC
(2) j = 0 to 7 for EVE3_EDMA_TPCC and EVE4_EDMA_TPCC
(3) k = 0 to 7 for EVE3_EDMA_TPCC and EVE4_EDMA_TPCC

Table 16-116. EVE3 and EVE4 EDMA_TPCC Registers Mapping Summary (L3_MAIN Access) (continued)

Register Name	Type	Register Width (Bits)	Address Offset	EVE3_EDMA_TPCC Physical Address (L3_MAIN Access)	EVE4_EDMA_TPCC Physical Address (L3_MAIN Access)
EDMA_TPCC_EMCR	W	32	0x0000 0308	0x422A 0308	0x423A 0308
EDMA_TPCC_EMCRH	W	32	0x0000 030C	0x422A 030C	0x423A 030C
EDMA_TPCC_QEMR	R	32	0x0000 0310	0x422A 0310	0x423A 0310
EDMA_TPCC_QEMCR	W	32	0x0000 0314	0x422A 0314	0x423A 0314
EDMA_TPCC_CCERR	R	32	0x0000 0318	0x422A 0318	0x423A 0318
EDMA_TPCC_CCERRCLR	W	32	0x0000 031C	0x422A 031C	0x423A 031C
EDMA_TPCC_EEVAL	W	32	0x0000 0320	0x422A 0320	0x423A 0320
EDMA_TPCC_DRAEM_k (3)	RW	32	0x0000 0340 + (0x8 * k)	-	-
EDMA_TPCC_DRAEHM_k (3)	RW	32	0x0000 0344 + (0x8 * k)	-	-
EDMA_TPCC_QRAEN_k (3)	RW	32	0x0000 0380 + (0x4 * k)	0x422A 0380 + (0x4 * k)	0x423A 0380 + (0x4 * k)
EDMA_TPCC_Q0E_p (4)	R	32	0x0000 0400 + (0x4 * p)	-	-
EDMA_TPCC_Q1E_p (4)	R	32	0x0000 0440 + (0x4 * p)	-	-
EDMA_TPCC_QSTATN_i (5)	R	32	0x0000 0600 + (0x4 * i)	0x422A 0600 + (0x4 * i)	0x423A 0600 + (0x4 * i)
EDMA_TPCC_QWMTHRA	RW	32	0x0000 0620	0x422A 0620	0x423A 0620
EDMA_TPCC_QWMTHRB	RW	32	0x0000 0624	0x422A 0624	0x423A 0624
EDMA_TPCC_CCSTAT	R	32	0x0000 0640	0x422A 0640	0x423A 0640
EDMA_TPCC_AETCTL	RW	32	0x0000 0700	0x422A 0700	0x423A 0700
EDMA_TPCC_AETSTAT	R	32	0x0000 0704	0x422A 0704	0x423A 0704
EDMA_TPCC_AETCMD	W	32	0x0000 0708	0x422A 0708	0x423A 0708
EDMA_TPCC_MPFAR	R	32	0x0000 0800	0x422A 0800	0x423A 0800
EDMA_TPCC_MPFSR	R	32	0x0000 0804	0x422A 0804	0x423A 0804
EDMA_TPCC_MPFCR	W	32	0x0000 0808	0x422A 0808	0x423A 0808
EDMA_TPCC_MPPAG	RW	32	0x0000 080C	0x422A 080C	0x423A 080C
EDMA_TPCC_MPPAN_k (6)	RW	32	0x0000 0810 + (0x4 * k)	0x422A 0810 + (0x4 * k)	0x423A 0810 + (0x4 * k)
EDMA_TPCC_ER	R	32	0x0000 1000	0x422A 1000	0x423A 1000
EDMA_TPCC_ERH	R	32	0x0000 1004	0x422A 1004	0x423A 1004
EDMA_TPCC_ECR	W	32	0x0000 1008	0x422A 1008	0x423A 1008
EDMA_TPCC_ECRH	W	32	0x0000 100C	0x422A 100C	0x423A 100C
EDMA_TPCC_ESR	W	32	0x0000 1010	0x422A 1010	0x423A 1010
EDMA_TPCC_ESRH	W	32	0x0000 1014	0x422A 1014	0x423A 1014
EDMA_TPCC_CER	R	32	0x0000 1018	0x422A 1018	0x423A 1018
EDMA_TPCC_CERH	R	32	0x0000 101C	0x422A 101C	0x423A 101C
EDMA_TPCC_EER	R	32	0x0000 1020	0x422A 1020	0x423A 1020
EDMA_TPCC_EERH	R	32	0x0000 1024	0x422A 1024	0x423A 1024
EDMA_TPCC_EECR	W	32	0x0000 1028	0x422A 1028	0x423A 1028
EDMA_TPCC_EECRH	W	32	0x0000 102C	0x422A 102C	0x423A 102C
EDMA_TPCC_EESR	W	32	0x0000 1030	0x422A 1030	0x423A 1030
EDMA_TPCC_EESRH	W	32	0x0000 1034	0x422A 1034	0x423A 1034
EDMA_TPCC_SER	R	32	0x0000 1038	0x422A 1038	0x423A 1038

(4) p = 0 to 15 for EVE3_EDMA_TPCC and EVE4_EDMA_TPCC
(5) i = 0 to 1 for EVE3_EDMA_TPCC and EVE4_EDMA_TPCC
(6) k = 0 to 7 for EVE3_EDMA_TPCC and EVE4_EDMA_TPCC

Table 16-116. EVE3 and EVE4 EDMA_TPCC Registers Mapping Summary (L3_MAIN Access) (continued)

Register Name	Type	Register Width (Bits)	Address Offset	EVE3_EDMA_TPCC Physical Address (L3_MAIN Access)	EVE4_EDMA_TPCC Physical Address (L3_MAIN Access)
EDMA_TPCC_SERH	R	32	0x0000 103C	0x422A 103C	0x423A 103C
EDMA_TPCC_SECR	W	32	0x0000 1040	0x422A 1040	0x423A 1040
EDMA_TPCC_SECRH	W	32	0x0000 1044	0x422A 1044	0x423A 1044
EDMA_TPCC_IER	R	32	0x0000 1050	0x422A 1050	0x423A 1050
EDMA_TPCC_IERH	R	32	0x0000 1054	0x422A 1054	0x423A 1054
EDMA_TPCC_IECR	W	32	0x0000 1058	0x422A 1058	0x423A 1058
EDMA_TPCC_IECRH	W	32	0x0000 105C	0x422A 105C	0x423A 105C
EDMA_TPCC_IESR	W	32	0x0000 1060	0x422A 1060	0x423A 1060
EDMA_TPCC_IESRH	W	32	0x0000 1064	0x422A 1064	0x423A 1064
EDMA_TPCC_IPR	R	32	0x0000 1068	0x422A 1068	0x423A 1068
EDMA_TPCC_IPRH	R	32	0x0000 106C	0x422A 106C	0x423A 106C
EDMA_TPCC_ICR	W	32	0x0000 1070	0x422A 1070	0x423A 1070
EDMA_TPCC_ICRH	W	32	0x0000 1074	0x422A 1074	0x423A 1074
EDMA_TPCC_IEVAL	W	32	0x0000 1078	0x422A 1078	0x423A 1078
EDMA_TPCC_QER	R	32	0x0000 1080	0x422A 1080	0x423A 1080
EDMA_TPCC_QEER	R	32	0x0000 1084	0x422A 1084	0x423A 1084
EDMA_TPCC_QEECR	W	32	0x0000 1088	0x422A 1088	0x423A 1088
EDMA_TPCC_QEESR	W	32	0x0000 108C	0x422A 108C	0x423A 108C
EDMA_TPCC_QSER	R	32	0x0000 1090	0x422A 1090	0x423A 1090
EDMA_TPCC_QSECR	W	32	0x0000 1094	0x422A 1094	0x423A 1094
EDMA_TPCC_ER_RN_k (6)	R	32	0x0000 2000 + (0x200 * k)	0x422A 2000 + (0x200 * k)	0x423A 2000 + (0x200 * k)
EDMA_TPCC_ERH_RN_k (6)	R	32	0x0000 2004 + (0x200 * k)	0x422A 2004 + (0x200 * k)	0x423A 2004 + (0x200 * k)
EDMA_TPCC_ECR_RN_k (6)	W	32	0x0000 2008 + (0x200 * k)	0x422A 2008 + (0x200 * k)	0x423A 2008 + (0x200 * k)
EDMA_TPCC_ECRH_RN_k (6)	W	32	0x0000 200C + (0x200 * k)	0x422A 200C + (0x200 * k)	0x423A 200C + (0x200 * k)
EDMA_TPCC_ESR_RN_k (7)	W	32	0x0000 2010 + (0x200 * k)	0x422A 2010 + (0x200 * k)	0x423A 2010 + (0x200 * k)
EDMA_TPCC_ESRH_RN_k (7)	W	32	0x0000 2014 + (0x200 * k)	0x422A 2014 + (0x200 * k)	0x423A 2014 + (0x200 * k)
EDMA_TPCC_CER_RN_k (7)	R	32	0x0000 2018 + (0x200 * k)	0x422A 2018 + (0x200 * k)	0x423A 2018 + (0x200 * k)
EDMA_TPCC_CERH_RN_k (7)	R	32	0x0000 201C + (0x200 * k)	0x422A 201C + (0x200 * k)	0x423A 201C + (0x200 * k)
EDMA_TPCC_EER_RN_k (7)	R	32	0x0000 2020 + (0x200 * k)	0x422A 2020 + (0x200 * k)	0x423A 2020 + (0x200 * k)
EDMA_TPCC_EERH_RN_k (7)	R	32	0x0000 2024 + (0x200 * k)	0x422A 2024 + (0x200 * k)	0x423A 2024 + (0x200 * k)
EDMA_TPCC_EECR_RN_k (7)	W	32	0x0000 2028 + (0x200 * k)	0x422A 2028 + (0x200 * k)	0x423A 2028 + (0x200 * k)
EDMA_TPCC_EECRH_RN_k (7)	W	32	0x0000 202C + (0x200 * k)	0x422A 202C + (0x200 * k)	0x423A 202C + (0x200 * k)
EDMA_TPCC_EESR_RN_k (7)	W	32	0x0000 2030 + (0x200 * k)	0x422A 2030 + (0x200 * k)	0x423A 2030 + (0x200 * k)
EDMA_TPCC_EESRH_RN_k (7)	W	32	0x0000 2034 + (0x200 * k)	0x422A 2034 + (0x200 * k)	0x423A 2034 + (0x200 * k)

(7) k = 0 to 7 for EVE3_EDMA_TPCC and EVE4_EDMA_TPCC

Table 16-116. EVE3 and EVE4 EDMA_TPCC Registers Mapping Summary (L3_MAIN Access) (continued)

Register Name	Type	Register Width (Bits)	Address Offset	EVE3_EDMA_TPCC Physical Address (L3_MAIN Access)	EVE4_EDMA_TPCC Physical Address (L3_MAIN Access)
EDMA_TPCC_SER_RN_k (7)	R	32	0x0000 2038 + (0x200 * k)	0x422A 2038 + (0x200 * k)	0x423A 2038 + (0x200 * k)
EDMA_TPCC_SERH_RN_k (7)	R	32	0x0000 203C + (0x200 * k)	0x422A 203C + (0x200 * k)	0x423A 203C + (0x200 * k)
EDMA_TPCC_SECR_RN_k (7)	W	32	0x0000 2040 + (0x200 * k)	0x422A 2040 + (0x200 * k)	0x423A 2040 + (0x200 * k)
EDMA_TPCC_SECRH_RN_k (7)	W	32	0x0000 2044 + (0x200 * k)	0x422A 2044 + (0x200 * k)	0x423A 2044 + (0x200 * k)
EDMA_TPCC_IER_RN_k (7)	R	32	0x0000 2050 + (0x200 * k)	0x422A 2050 + (0x200 * k)	0x423A 2050 + (0x200 * k)
EDMA_TPCC_IERH_RN_k (7)	R	32	0x0000 2054 + (0x200 * k)	0x422A 2054 + (0x200 * k)	0x423A 2054 + (0x200 * k)
EDMA_TPCC_IECR_RN_k (7)	W	32	0x0000 2058 + (0x200 * k)	0x422A 2058 + (0x200 * k)	0x423A 2058 + (0x200 * k)
EDMA_TPCC_IECRH_RN_k (7)	W	32	0x0000 205C + (0x200 * k)	0x422A 205C + (0x200 * k)	0x423A 205C + (0x200 * k)
EDMA_TPCC_IESR_RN_k (7)	W	32	0x0000 2060 + (0x200 * k)	0x422A 2060 + (0x200 * k)	0x423A 2060 + (0x200 * k)
EDMA_TPCC_IESRH_RN_k (7)	W	32	0x0000 2064 + (0x200 * k)	0x422A 2064 + (0x200 * k)	0x423A 2064 + (0x200 * k)
EDMA_TPCC_IPR_RN_k (7)	R	32	0x0000 2068 + (0x200 * k)	0x422A 2068 + (0x200 * k)	0x423A 2068 + (0x200 * k)
EDMA_TPCC_IPRH_RN_k (7)	R	32	0x0000 206C + (0x200 * k)	0x422A 206C + (0x200 * k)	0x423A 206C + (0x200 * k)
EDMA_TPCC_ICR_RN_k (7)	W	32	0x0000 2070 + (0x200 * k)	0x422A 2070 + (0x200 * k)	0x423A 2070 + (0x200 * k)
EDMA_TPCC_ICRH_RN_k (7)	W	32	0x0000 2074 + (0x200 * k)	0x422A 2074 + (0x200 * k)	0x423A 2074 + (0x200 * k)
EDMA_TPCC_IEVAL_RN_k (7)	W	32	0x0000 2078 + (0x200 * k)	0x422A 2078 + (0x200 * k)	0x423A 2078 + (0x200 * k)
EDMA_TPCC_QER_RN_k (7)	R	32	0x0000 2080 + (0x200 * k)	0x422A 2080 + (0x200 * k)	0x423A 2080 + (0x200 * k)
EDMA_TPCC_QEER_RN_k (8)	R	32	0x0000 2084 + (0x200 * k)	0x422A 2084 + (0x200 * k)	0x423A 2084 + (0x200 * k)
EDMA_TPCC_QEECR_RN_k (8)	W	32	0x0000 2088 + (0x200 * k)	0x422A 2088 + (0x200 * k)	0x423A 2088 + (0x200 * k)
EDMA_TPCC_QEESR_RN_k (8)	W	32	0x0000 208C + (0x200 * k)	0x422A 208C + (0x200 * k)	0x423A 208C + (0x200 * k)
EDMA_TPCC_QSER_RN_k (8)	R	32	0x0000 2090 + (0x200 * k)	0x422A 2090 + (0x200 * k)	0x423A 2090 + (0x200 * k)
EDMA_TPCC_QSECR_RN_k (8)	W	32	0x0000 2094 + (0x200 * k)	0x422A 2094 + (0x200 * k)	0x423A 2094 + (0x200 * k)
EDMA_TPCC_OPT_n (9)	RW	32	0x0000 4000 + (0x20 * n)	0x422A 4000 + (0x20 * n)	0x423A 4000 + (0x20 * n)
EDMA_TPCC_SRC_n (9)	RW	32	0x0000 4004 + (0x20 * n)	0x422A 4004 + (0x20 * n)	0x423A 4004 + (0x20 * n)
EDMA_TPCC_ABCNT_n (9)	RW	32	0x0000 4008 + (0x20 * n)	0x422A 4008 + (0x20 * n)	0x423A 4008 + (0x20 * n)
EDMA_TPCC_DST_n (9)	RW	32	0x0000 400C + (0x20 * n)	0x422A 400C + (0x20 * n)	0x423A 400C + (0x20 * n)
EDMA_TPCC_BIDX_n (9)	RW	32	0x0000 4010 + (0x20 * n)	0x422A 4010 + (0x20 * n)	0x423A 4010 + (0x20 * n)

(8) k = 0 to 7 for EVE3_EDMA_TPCC and EVE4_EDMA_TPCC
(9) n = 0 to 127 for EVE3_EDMA_TPCC and EVE4_EDMA_TPCC

Table 16-116. EVE3 and EVE4 EDMA_TPCC Registers Mapping Summary (L3_MAIN Access) (continued)

Register Name	Type	Register Width (Bits)	Address Offset	EVE3_EDMA_TPCC Physical Address (L3_MAIN Access)	EVE4_EDMA_TPCC Physical Address (L3_MAIN Access)
EDMA_TPCC_LNK_n [9]	RW	32	0x0000 4014 + (0x20 * n)	0x422A 4014 + (0x20 * n)	0x423A 4014 + (0x20 * n)
EDMA_TPCC_CIDX_n [9]	RW	32	0x0000 4018 + (0x20 * n)	0x422A 4018 + (0x20 * n)	0x423A 4018 + (0x20 * n)
EDMA_TPCC_CCNT_n [9]	RW	32	0x0000 401C + (0x20 * n)	0x422A 401C + (0x20 * n)	0x423A 401C + (0x20 * n)

NOTE: The value for "n" is from 0 to 1 in the Table 16-117. It corresponds of the Transfer Controller (EDMA_TPTC0 and EDMA_TPTC1) instances in the device.

Table 16-117. System EDMA TPTC0 and EDMA TPTC1 Registers Mapping Summary

Register Name	Type	Register Width (Bits)	Address Offset	SYS_EDMA_TPTC0 Physical Address (L3_MAIN Access)	SYS_EDMA_TPTC1 Physical Address (L3_MAIN Access)
EDMA_TPTCn_PID	R	32	0x0000 0000	0x4340 0000	0x4350 0000
EDMA_TPTCn_TCCFG	R	32	0x0000 0004	0x4340 0004	0x4350 0004
EDMA_TPTCn_TCSTAT	R	32	0x0000 0100	0x4340 0100	0x4350 0100
EDMA_TPTCn_INTSTAT	R	32	0x0000 0104	0x4340 0104	0x4350 0104
EDMA_TPTCn_INTEN	RW	32	0x0000 0108	0x4340 0108	0x4350 0108
EDMA_TPTCn_INTCLR	W	32	0x0000 010C	0x4340 010C	0x4350 010C
EDMA_TPTCn_INTCMD	W	32	0x0000 0110	0x4340 0110	0x4350 0110
EDMA_TPTCn_ERRSTAT	R	32	0x0000 0120	0x4340 0120	0x4350 0120
EDMA_TPTCn_ERREN	RW	32	0x0000 0124	0x4340 0124	0x4350 0124
EDMA_TPTCn_ERRCLR	W	32	0x0000 0128	0x4340 0128	0x4350 0128
EDMA_TPTCn_ERRDET	R	32	0x0000 012C	0x4340 012C	0x4350 012C
EDMA_TPTCn_ERRCMD	W	32	0x0000 0130	0x4340 0130	0x4350 0130
EDMA_TPTCn_RDRATE	RW	32	0x0000 0140	0x4340 0140	0x4350 0140
EDMA_TPTCn_POPT	RW	32	0x0000 0200	0x4340 0200	0x4350 0200
EDMA_TPTCn_PSRC	RW	32	0x0000 0204	0x4340 0204	0x4350 0204
EDMA_TPTCn_PCNT	RW	32	0x0000 0208	0x4340 0208	0x4350 0208
EDMA_TPTCn_PDST	RW	32	0x0000 020C	0x4340 020C	0x4350 020C
EDMA_TPTCn_PBIDX	RW	32	0x0000 0210	0x4340 0210	0x4350 0210
EDMA_TPTCn_PMPPRXY	R	32	0x0000 0214	0x4340 0214	0x4350 0214
EDMA_TPTCn_SAOPT	R	32	0x0000 0240	0x4340 0240	0x4350 0240
EDMA_TPTCn_SASRC	R	32	0x0000 0244	0x4340 0244	0x4350 0244
EDMA_TPTCn_SACNT	R	32	0x0000 0248	0x4340 0248	0x4350 0248
EDMA_TPTCn_SADST	R	32	0x0000 024C	0x4340 024C	0x4350 024C
EDMA_TPTCn_SABIDX	R	32	0x0000 0250	0x4340 0250	0x4350 0250
EDMA_TPTCn_SAMPPRXY	R	32	0x0000 0254	0x4340 0254	0x4350 0254
EDMA_TPTCn_SACNTRLD	R	32	0x0000 0258	0x4340 0258	0x4350 0258
EDMA_TPTCn_SASRCBREF	R	32	0x0000 025C	0x4340 025C	0x4350 025C
EDMA_TPTCn_SADSTBREF	R	32	0x0000 0260	0x4340 0260	0x4350 0260
EDMA_TPTCn_DFCNTRLD	R	32	0x0000 0280	0x4340 0280	0x4350 0280
EDMA_TPTCn_DFSRCBREF	R	32	0x0000 0284	0x4340 0284	0x4350 0284
EDMA_TPTCn_DFDSTBREF	R	32	0x0000 0288	0x4340 0288	0x4350 0288

Table 16-117. System EDMA TPTC0 and EDMA TPTC1 Registers Mapping Summary (continued)

Register Name	Type	Register Width (Bits)	Address Offset	SYS_EDMA_TPTC0 Physical Address (L3_MAIN Access)	SYS_EDMA_TPTC1 Physical Address (L3_MAIN Access)
EDMA_TPTCn_DFOPTi [(1)]	R	32	0x0000 0300 + (0x40 * i)	0x4340 0300 + (0x40 * i)	0x4350 0300 + (0x40 * i)
EDMA_TPTCn_DFSRCi [(1)]	R	32	0x0000 0304 + (0x40 * i)	0x4340 0304 + (0x40 * i)	0x4350 0304 + (0x40 * i)
EDMA_TPTCn_DFCNTi [(1)]	R	32	0x0000 0308 + (0x40 * i)	0x4340 0308 + (0x40 * i)	0x4350 0308 + (0x40 * i)
EDMA_TPTCn_DFDSTi [(1)]	R	32	0x0000 030C + (0x40 * i)	0x4340 030C + (0x40 * i)	0x4350 030C + (0x40 * i)
EDMA_TPTCn_DFBIDXi [(1)]	R	32	0x0000 0310 + (0x40 * i)	0x4340 0310 + (0x40 * i)	0x4350 0310 + (0x40 * i)
EDMA_TPTCn_DFMPPRXYi [(1)]	R	32	0x0000 0314 + (0x40 * i)	0x4340 0314 + (0x40 * i)	0x4350 0314 + (0x40 * i)

(1) i = 0 to 1 for SYS_EDMA_TPTC0 and SYS_EDMA_TPTC1

Table 16-118. DSP1 EDMA_TPTC0 and EDMA_TPTC1 Registers Mapping Summary (L3_MAIN Access)

Register Name	Type	Register Width (Bits)	Address Offset	DSP1_EDMA_TPTC0 Physical Address (L3_MAIN Access)	DSP1_EDMA_TPTC1 Physical Address (L3_MAIN Access)
EDMA_TPTCn_PID	R	32	0x0000 0000	0x40D0 5000	0x40D0 6000
EDMA_TPTCn_TCCFG	R	32	0x0000 0004	0x40D0 5004	0x40D0 6004
EDMA_TPTCn_TCSTAT	R	32	0x0000 0100	0x40D0 5100	0x40D0 6100
EDMA_TPTCn_INTSTAT	R	32	0x0000 0104	0x40D0 5104	0x40D0 6104
EDMA_TPTCn_INTEN	RW	32	0x0000 0108	0x40D0 5108	0x40D0 6108
EDMA_TPTCn_INTCLR	W	32	0x0000 010C	0x40D0 510C	0x40D0 610C
EDMA_TPTCn_INTCMD	W	32	0x0000 0110	0x40D0 5110	0x40D0 6110
EDMA_TPTCn_ERRSTAT	R	32	0x0000 0120	0x40D0 5120	0x40D0 6120
EDMA_TPTCn_ERREN	RW	32	0x0000 0124	0x40D0 5124	0x40D0 6124
EDMA_TPTCn_ERRCLR	W	32	0x0000 0128	0x40D0 5128	0x40D0 6128
EDMA_TPTCn_ERRDET	R	32	0x0000 012C	0x40D0 512C	0x40D0 612C
EDMA_TPTCn_ERRCMD	W	32	0x0000 0130	0x40D0 5130	0x40D0 6130
EDMA_TPTCn_RDRATE	RW	32	0x0000 0140	0x40D0 5140	0x40D0 6140
EDMA_TPTCn_POPT	RW	32	0x0000 0200	0x40D0 5200	0x40D0 6200
EDMA_TPTCn_PSRC	RW	32	0x0000 0204	0x40D0 5204	0x40D0 6204
EDMA_TPTCn_PCNT	RW	32	0x0000 0208	0x40D0 5208	0x40D0 6208
EDMA_TPTCn_PDST	RW	32	0x0000 020C	0x40D0 520C	0x40D0 620C
EDMA_TPTCn_PBIDX	RW	32	0x0000 0210	0x40D0 5210	0x40D0 6210
EDMA_TPTCn_PMPPRXY	R	32	0x0000 0214	0x40D0 5214	0x40D0 6214
EDMA_TPTCn_SAOPT	R	32	0x0000 0240	0x40D0 5240	0x40D0 6240
EDMA_TPTCn_SASRC	R	32	0x0000 0244	0x40D0 5244	0x40D0 6244
EDMA_TPTCn_SACNT	R	32	0x0000 0248	0x40D0 5248	0x40D0 6248
EDMA_TPTCn_SADST	R	32	0x0000 024C	0x40D0 524C	0x40D0 624C
EDMA_TPTCn_SABIDX	R	32	0x0000 0250	0x40D0 5250	0x40D0 6250
EDMA_TPTCn_SAMPPRXY	R	32	0x0000 0254	0x40D0 5254	0x40D0 6254
EDMA_TPTCn_SACNTRLD	R	32	0x0000 0258	0x40D0 5258	0x40D0 6258
EDMA_TPTCn_SASRCBREF	R	32	0x0000 025C	0x40D0 525C	0x40D0 625C
EDMA_TPTCn_SADSTBREF	R	32	0x0000 0260	0x40D0 5260	0x40D0 6260
EDMA_TPTCn_DFCNTRLD	R	32	0x0000 0280	0x40D0 5280	0x40D0 6280

Table 16-118. DSP1 EDMA_TPTC0 and EDMA_TPTC1 Registers Mapping Summary (L3_MAIN Access) (continued)

Register Name	Type	Register Width (Bits)	Address Offset	DSP1_EDMA_TPTC0 Physical Address (L3_MAIN Access)	DSP1_EDMA_TPTC1 Physical Address (L3_MAIN Access)
EDMA_TPTCn_DFSRCBREF	R	32	0x0000 0284	0x40D0 5284	0x40D0 6284
EDMA_TPTCn_DFDSTBREF	R	32	0x0000 0288	0x40D0 5288	0x40D0 6288
EDMA_TPTCn_DFOPTi[(1)]	R	32	0x0000 0300 + (0x40 * i)	0x40D0 5300 + (0x40 * i)	0x40D0 6300 + (0x40 * i)
EDMA_TPTCn_DFSRCi [(1)]	R	32	0x0000 0304 + (0x40 * i)	0x40D0 5304 + (0x40 * i)	0x40D0 6304 + (0x40 * i)
EDMA_TPTCn_DFCNTi [(1)]	R	32	0x0000 0308 + (0x40 * i)	0x40D0 5308 + (0x40 * i)	0x40D0 6308 + (0x40 * i)
EDMA_TPTCn_DFDSTi [(1)]	R	32	0x0000 030C + (0x40 * i)	0x40D0 530C + (0x40 * i)	0x40D0 630C + (0x40 * i)
EDMA_TPTCn_DFBIDXi [(1)]	R	32	0x0000 0310 + (0x40 * i)	0x40D0 5310 + (0x40 * i)	0x40D0 6310 + (0x40 * i)
EDMA_TPTCn_DFMPPRXYi [(1)]	R	32	0x0000 0314 + (0x40 * i)	0x40D0 5314 + (0x40 * i)	0x40D0 6314 + (0x40 * i)

(1) i = 0 to 1 for DSP1_EDMA_TPTC0 and DSP1_EDMA_TPTC1

Table 16-119. DSP2 EDMA_TPTC0 and EDMA_TPTC1 Registers Mapping Summary (L3_MAIN Access)

Register Name	Type	Register Width (Bits)	Address Offset	DSP2_EDMA_TPTC0 Physical Address (L3_MAIN Access)	DSP2_EDMA_TPTC1 Physical Address (L3_MAIN Access)
EDMA_TPTCn_PID	R	32	0x0000 0000	0x4150 5000	0x4150 6000
EDMA_TPTCn_TCCFG	R	32	0x0000 0004	0x4150 5004	0x4150 6004
EDMA_TPTCn_TCSTAT	R	32	0x0000 0100	0x4150 5100	0x4150 6100
EDMA_TPTCn_INTSTAT	R	32	0x0000 0104	0x4150 5104	0x4150 6104
EDMA_TPTCn_INTEN	RW	32	0x0000 0108	0x4150 5108	0x4150 6108
EDMA_TPTCn_INTCLR	W	32	0x0000 010C	0x4150 510C	0x4150 610C
EDMA_TPTCn_INTCMD	W	32	0x0000 0110	0x4150 5110	0x4150 6110
EDMA_TPTCn_ERRSTAT	R	32	0x0000 0120	0x4150 5120	0x4150 6120
EDMA_TPTCn_ERREN	RW	32	0x0000 0124	0x4150 5124	0x4150 6124
EDMA_TPTCn_ERRCLR	W	32	0x0000 0128	0x4150 5128	0x4150 6128
EDMA_TPTCn_ERRDET	R	32	0x0000 012C	0x4150 512C	0x4150 612C
EDMA_TPTCn_ERRCMD	W	32	0x0000 0130	0x4150 5130	0x4150 6130
EDMA_TPTCn_RDRATE	RW	32	0x0000 0140	0x4150 5140	0x4150 6140
EDMA_TPTCn_POPT	RW	32	0x0000 0200	0x4150 5200	0x4150 6200
EDMA_TPTCn_PSRC	RW	32	0x0000 0204	0x4150 5204	0x4150 6204
EDMA_TPTCn_PCNT	RW	32	0x0000 0208	0x4150 5208	0x4150 6208
EDMA_TPTCn_PDST	RW	32	0x0000 020C	0x4150 520C	0x4150 620C
EDMA_TPTCn_PBIDX	RW	32	0x0000 0210	0x4150 5210	0x4150 6210
EDMA_TPTCn_PMPPRXY	R	32	0x0000 0214	0x4150 5214	0x4150 6214
EDMA_TPTCn_SAOPT	R	32	0x0000 0240	0x4150 5240	0x4150 6240
EDMA_TPTCn_SASRC	R	32	0x0000 0244	0x4150 5244	0x4150 6244
EDMA_TPTCn_SACNT	R	32	0x0000 0248	0x4150 5248	0x4150 6248
EDMA_TPTCn_SADST	R	32	0x0000 024C	0x4150 524C	0x4150 624C
EDMA_TPTCn_SABIDX	R	32	0x0000 0250	0x4150 5250	0x4150 6250
EDMA_TPTCn_SAMPPRXY	R	32	0x0000 0254	0x4150 5254	0x4150 6254
EDMA_TPTCn_SACNTRLD	R	32	0x0000 0258	0x4150 5258	0x4150 6258

Table 16-119. DSP2 EDMA_TPTC0 and EDMA_TPTC1 Registers Mapping Summary (L3_MAIN Access) (continued)

Register Name	Type	Register Width (Bits)	Address Offset	DSP2_EDMA_TPTC0 Physical Address (L3_MAIN Access)	DSP2_EDMA_TPTC1 Physical Address (L3_MAIN Access)
EDMA_TPTCn_SASRCBREF	R	32	0x0000 025C	0x4150 525C	0x4150 625C
EDMA_TPTCn_SADSTBREF	R	32	0x0000 0260	0x4150 5260	0x4150 6260
EDMA_TPTCn_DFCNTRLD	R	32	0x0000 0280	0x4150 5280	0x4150 6280
EDMA_TPTCn_DFSRCBREF	R	32	0x0000 0284	0x4150 5284	0x4150 6284
EDMA_TPTCn_DFDSTBREF	R	32	0x0000 0288	0x4150 5288	0x4150 6288
EDMA_TPTCn_DFOPTi [(1)]	R	32	0x0000 0300 + (0x40 * i)	0x4150 5300 + (0x40 * i)	0x4150 6300 + (0x40 * i)
EDMA_TPTCn_DFSRCi [(1)]	R	32	0x0000 0304 + (0x40 * i)	0x4150 5304 + (0x40 * i)	0x4150 6304 + (0x40 * i)
EDMA_TPTCn_DFCNTi [(1)]	R	32	0x0000 0308 + (0x40 * i)	0x4150 5308 + (0x40 * i)	0x4150 6308 + (0x40 * i)
EDMA_TPTCn_DFDSTi [(1)]	R	32	0x0000 030C + (0x40 * i)	0x4150 530C + (0x40 * i)	0x4150 630C + (0x40 * i)
EDMA_TPTCn_DFBIDXi [(1)]	R	32	0x0000 0310 + (0x40 * i)	0x4150 5310 + (0x40 * i)	0x4150 6310 + (0x40 * i)
EDMA_TPTCn_DFMPPRXYi [(1)]	R	32	0x0000 0314 + (0x40 * i)	0x4150 5314 + (0x40 * i)	0x4150 6314 + (0x40 * i)

(1) i = 0 to 1 for DSP2_EDMA_TPTC0 and DSP2_EDMA_TPTC1

Table 16-120. DSP EDMA_TPTC0 and EDMA_TPTC1 Registers Mapping Summary (Private Access)

Register Name	Type	Register Width (Bits)	Address Offset	DSP_EDMA_TPTC0 Physical Address (DSP Private Access)	DSP_EDMA_TPTC1 Physical Address (DSP Private Access)
EDMA_TPTCn_PID	R	32	0x0000 0000	0x01D0 5000	0x01D0 6000
EDMA_TPTCn_TCCFG	R	32	0x0000 0004	0x01D0 5004	0x01D0 6004
EDMA_TPTCn_TCSTAT	R	32	0x0000 0100	0x01D0 5100	0x01D0 6100
EDMA_TPTCn_INTSTAT	R	32	0x0000 0104	0x01D0 5104	0x01D0 6104
EDMA_TPTCn_INTEN	RW	32	0x0000 0108	0x01D0 5108	0x01D0 6108
EDMA_TPTCn_INTCLR	W	32	0x0000 010C	0x01D0 510C	0x01D0 610C
EDMA_TPTCn_INTCMD	W	32	0x0000 0110	0x01D0 5110	0x01D0 6110
EDMA_TPTCn_ERRSTAT	R	32	0x0000 0120	0x01D0 5120	0x01D0 6120
EDMA_TPTCn_ERREN	RW	32	0x0000 0124	0x01D0 5124	0x01D0 6124
EDMA_TPTCn_ERRCLR	W	32	0x0000 0128	0x01D0 5128	0x01D0 6128
EDMA_TPTCn_ERRDET	R	32	0x0000 012C	0x01D0 512C	0x01D0 612C
EDMA_TPTCn_ERRCMD	W	32	0x0000 0130	0x01D0 5130	0x01D0 6130
EDMA_TPTCn_RDRATE	RW	32	0x0000 0140	0x01D0 5140	0x01D0 6140
EDMA_TPTCn_POPT	RW	32	0x0000 0200	0x01D0 5200	0x01D0 6200
EDMA_TPTCn_PSRC	RW	32	0x0000 0204	0x01D0 5204	0x01D0 6204
EDMA_TPTCn_PCNT	RW	32	0x0000 0208	0x01D0 5208	0x01D0 6208
EDMA_TPTCn_PDST	RW	32	0x0000 020C	0x01D0 520C	0x01D0 620C
EDMA_TPTCn_PBIDX	RW	32	0x0000 0210	0x01D0 5210	0x01D0 6210
EDMA_TPTCn_PMPPRXY	R	32	0x0000 0214	0x01D0 5214	0x01D0 6214
EDMA_TPTCn_SAOPT	R	32	0x0000 0240	0x01D0 5240	0x01D0 6240
EDMA_TPTCn_SASRC	R	32	0x0000 0244	0x01D0 5244	0x01D0 6244
EDMA_TPTCn_SACNT	R	32	0x0000 0248	0x01D0 5248	0x01D0 6248
EDMA_TPTCn_SADST	R	32	0x0000 024C	0x01D0 524C	0x01D0 624C
EDMA_TPTCn_SABIDX	R	32	0x0000 0250	0x01D0 5250	0x01D0 6250

Table 16-120. DSP EDMA_TPTC0 and EDMA_TPTC1 Registers Mapping Summary (Private Access) (continued)

Register Name	Type	Register Width (Bits)	Address Offset	DSP_EDMA_TPTC0 Physical Address (DSP Private Access)	DSP_EDMA_TPTC1 Physical Address (DSP Private Access)
EDMA_TPTCn_SAMPPRXY	R	32	0x0000 0254	0x01D0 5254	0x01D0 6254
EDMA_TPTCn_SACNTRLD	R	32	0x0000 0258	0x01D0 5258	0x01D0 6258
EDMA_TPTCn_SASRCBREF	R	32	0x0000 025C	0x01D0 525C	0x01D0 625C
EDMA_TPTCn_SADSTBREF	R	32	0x0000 0260	0x01D0 5260	0x01D0 6260
EDMA_TPTCn_DFCNTRLD	R	32	0x0000 0280	0x01D0 5280	0x01D0 6280
EDMA_TPTCn_DFSRCBREF	R	32	0x0000 0284	0x01D0 5284	0x01D0 6284
EDMA_TPTCn_DFDSTBREF	R	32	0x0000 0288	0x01D0 5288	0x01D0 6288
EDMA_TPTCn_DFOPTi [(1)]	R	32	0x0000 0300 + (0x40 * i)	0x01D0 5300 + (0x40 * i)	0x01D0 6300 + (0x40 * i)
EDMA_TPTCn_DFSRCi [(1)]	R	32	0x0000 0304 + (0x40 * i)	0x01D0 5304 + (0x40 * i)	0x01D0 6304 + (0x40 * i)
EDMA_TPTCn_DFCNTi [(1)]	R	32	0x0000 0308 + (0x40 * i)	0x01D0 5308 + (0x40 * i)	0x01D0 6308 + (0x40 * i)
EDMA_TPTCn_DFDSTi [(1)]	R	32	0x0000 030C + (0x40 * i)	0x01D0 530C + (0x40 * i)	0x01D0 630C + (0x40 * i)
EDMA_TPTCn_DFBIDXi [(1)]	R	32	0x0000 0310 + (0x40 * i)	0x01D0 5310 + (0x40 * i)	0x01D0 6310 + (0x40 * i)
EDMA_TPTCn_DFMPPRXYi [(1)]	R	32	0x0000 0314 + (0x40 * i)	0x01D0 5314 + (0x40 * i)	0x01D0 6314 + (0x40 * i)

(1) i = 0 to 1 for DSP_EDMA_TPTC0 and DSP_EDMA_TPTC1

NOTE: EVE is not supported in this family of devices.

Table 16-121. EVE1 and EVE2 EDMA_TPTC0 Registers Mapping Summary (L3_MAIN Access)

Register Name	Type	Register Width (Bits)	Address Offset	EVE1_EDMA_TPTC0 Physical Address (L3_MAIN Access)	EVE2_EDMA_TPTC0 Physical Address (L3_MAIN Access)
EDMA_TPTCn_PID	R	32	0x0000 0000	-	-
EDMA_TPTCn_TCCFG	R	32	0x0000 0004	-	-
EDMA_TPTCn_TCSTAT	R	32	0x0000 0100	0x4208 6100	0x4218 6100
EDMA_TPTCn_INTSTAT	R	32	0x0000 0104	0x4208 6104	0x4218 6104
EDMA_TPTCn_INTEN	RW	32	0x0000 0108	0x4208 6108	0x4218 6108
EDMA_TPTCn_INTCLR	W	32	0x0000 010C	0x4208 610C	0x4218 610C
EDMA_TPTCn_INTCMD	W	32	0x0000 0110	0x4208 6110	0x4218 6110
EDMA_TPTCn_ERRSTAT	R	32	0x0000 0120	0x4208 6120	0x4218 6120
EDMA_TPTCn_ERREN	RW	32	0x0000 0124	0x4208 6124	0x4218 6124
EDMA_TPTCn_ERRCLR	W	32	0x0000 0128	0x4208 6128	0x4218 6128
EDMA_TPTCn_ERRDET	R	32	0x0000 012C	0x4208 612C	0x4218 612C
EDMA_TPTCn_ERRCMD	W	32	0x0000 0130	0x4208 6130	0x4218 6130
EDMA_TPTCn_RDRATE	RW	32	0x0000 0140	0x4208 6140	0x4218 6140
EDMA_TPTCn_POPT	RW	32	0x0000 0200	0x4208 6200	0x4218 6200
EDMA_TPTCn_PSRC	RW	32	0x0000 0204	0x4208 6204	0x4218 6204
EDMA_TPTCn_PCNT	RW	32	0x0000 0208	0x4208 6208	0x4218 6208
EDMA_TPTCn_PDST	RW	32	0x0000 020C	0x4208 620C	0x4218 620C
EDMA_TPTCn_PBIDX	RW	32	0x0000 0210	0x4208 6210	0x4218 6210
EDMA_TPTCn_PMPPRXY	R	32	0x0000 0214	0x4208 6214	0x4218 6214

Table 16-121. EVE1 and EVE2 EDMA_TPTC0 Registers Mapping Summary (L3_MAIN Access) (continued)

Register Name	Type	Register Width (Bits)	Address Offset	EVE1_EDMA_TPTC0 Physical Address (L3_MAIN Access)	EVE2_EDMA_TPTC0 Physical Address (L3_MAIN Access)
EDMA_TPTCn_SAOPT	R	32	0x0000 0240	0x4208 6240	0x4218 6240
EDMA_TPTCn_SASRC	R	32	0x0000 0244	0x4208 6244	0x4218 6244
EDMA_TPTCn_SACNT	R	32	0x0000 0248	0x4208 6248	0x4218 6248
EDMA_TPTCn_SADST	R	32	0x0000 024C	-	-
EDMA_TPTCn_SABIDX	R	32	0x0000 0250	0x4208 6250	0x4218 6250
EDMA_TPTCn_SAMPPRXY	R	32	0x0000 0254	0x4208 6254	0x4218 6254
EDMA_TPTCn_SACNTRLD	R	32	0x0000 0258	0x4208 6258	0x4218 6258
EDMA_TPTCn_SASRCBREF	R	32	0x0000 025C	0x4208 625C	0x4218 625C
EDMA_TPTCn_SADSTBREF	R	32	0x0000 0260	0x4208 6260	0x4218 6260
EDMA_TPTCn_DFCNTRLD	R	32	0x0000 0280	0x4208 6280	0x4218 6280
EDMA_TPTCn_DFSRCBREF	R	32	0x0000 0284	0x4208 6284	0x4218 6284
EDMA_TPTCn_DFDSTBREF	R	32	0x0000 0288	-	-
EDMA_TPTCn_DFOPTi (1)	R	32	0x0000 0300 + (0x40 * i)	0x4208 6300 + (0x40 * i)	0x4218 6300 + (0x40 * i)
EDMA_TPTCn_DFSRCi (1)	R	32	0x0000 0304 + (0x40 * i)	0x4208 6304 + (0x40 * i)	0x4218 6304 + (0x40 * i)
EDMA_TPTCn_DFCNTi (1)	R	32	0x0000 0308 + (0x40 * i)	0x4208 6308 + (0x40 * i)	0x4218 6308 + (0x40 * i)
EDMA_TPTCn_DFDSTi (1)	R	32	0x0000 030C + (0x40 * i)	0x4208 630C + (0x40 * i)	0x4218 630C + (0x40 * i)
EDMA_TPTCn_DFBIDXi (1)	R	32	0x0000 0310 + (0x40 * i)	0x4208 6310 + (0x40 * i)	0x4218 6310 + (0x40 * i)
EDMA_TPTCn_DFMPPRXYi (1)	R	32	0x0000 0314 + (0x40 * i)	0x4208 6314 + (0x40 * i)	0x4218 6314 + (0x40 * i)

(1) i = 0 to 1 for EVE1_EDMA_TPTC0 and EVE2_EDMA_TPTC0

NOTE: EVE is not supported in this family of devices.

Table 16-122. EVE1 and EVE2 EDMA_TPTC1 Registers Mapping Summary (L3_MAIN Access)

Register Name	Type	Register Width (Bits)	Address Offset	EVE1_EDMA_TPTC1 Physical Address (L3_MAIN Access)	EVE2_EDMA_TPTC1 Physical Address (L3_MAIN Access)
EDMA_TPTCn_PID	R	32	0x0000 0000	-	-
EDMA_TPTCn_TCCFG	R	32	0x0000 0004	-	-
EDMA_TPTCn_TCSTAT	R	32	0x0000 0100	0x4208 7100	0x4218 7100
EDMA_TPTCn_INTSTAT	R	32	0x0000 0104	0x4208 7104	0x4218 7104
EDMA_TPTCn_INTEN	RW	32	0x0000 0108	0x4208 7108	0x4218 7108
EDMA_TPTCn_INTCLR	W	32	0x0000 010C	0x4208 710C	0x4218 710C
EDMA_TPTCn_INTCMD	W	32	0x0000 0110	0x4208 7110	0x4218 7110
EDMA_TPTCn_ERRSTAT	R	32	0x0000 0120	0x4208 7120	0x4218 7120
EDMA_TPTCn_ERREN	RW	32	0x0000 0124	0x4208 7124	0x4218 7124
EDMA_TPTCn_ERRCLR	W	32	0x0000 0128	0x4208 7128	0x4218 7128
EDMA_TPTCn_ERRDET	R	32	0x0000 012C	0x4208 712C	0x4218 712C
EDMA_TPTCn_ERRCMD	W	32	0x0000 0130	0x4208 7130	0x4218 7130
EDMA_TPTCn_RDRATE	RW	32	0x0000 0140	0x4208 7140	0x4218 7140
EDMA_TPTCn_POPT	RW	32	0x0000 0200	0x4208 7200	0x4218 7200

Table 16-122. EVE1 and EVE2 EDMA_TPTC1 Registers Mapping Summary (L3_MAIN Access) (continued)

Register Name	Type	Register Width (Bits)	Address Offset	EVE1_EDMA_TPTC1 Physical Address (L3_MAIN Access)	EVE2_EDMA_TPTC1 Physical Address (L3_MAIN Access)
EDMA_TPTCn_PSRC	RW	32	0x0000 0204	0x4208 7204	0x4218 7204
EDMA_TPTCn_PCNT	RW	32	0x0000 0208	0x4208 7208	0x4218 7208
EDMA_TPTCn_PDST	RW	32	0x0000 020C	0x4208 720C	0x4218 720C
EDMA_TPTCn_PBIDX	RW	32	0x0000 0210	0x4208 7210	0x4218 7210
EDMA_TPTCn_PMPPRXY	R	32	0x0000 0214	0x4208 7214	0x4218 7214
EDMA_TPTCn_SAOPT	R	32	0x0000 0240	0x4208 7240	0x4218 7240
EDMA_TPTCn_SASRC	R	32	0x0000 0244	0x4208 7244	0x4218 7244
EDMA_TPTCn_SACNT	R	32	0x0000 0248	0x4208 7248	0x4218 7248
EDMA_TPTCn_SADST	R	32	0x0000 024C	-	-
EDMA_TPTCn_SABIDX	R	32	0x0000 0250	0x4208 7250	0x4218 7250
EDMA_TPTCn_SAMPPRXY	R	32	0x0000 0254	0x4208 7254	0x4218 7254
EDMA_TPTCn_SACNTRLD	R	32	0x0000 0258	0x4208 7258	0x4218 7258
EDMA_TPTCn_SASRCBREF	R	32	0x0000 025C	0x4208 725C	0x4218 725C
EDMA_TPTCn_SADSTBREF	R	32	0x0000 0260	0x4208 7260	0x4218 7260
EDMA_TPTCn_DFCNTRLD	R	32	0x0000 0280	0x4208 7280	0x4218 7280
EDMA_TPTCn_DFSRCBREF	R	32	0x0000 0284	0x4208 7284	0x4218 7284
EDMA_TPTCn_DFDSTBREF	R	32	0x0000 0288	-	-
EDMA_TPTCn_DFOPTi(1)	R	32	0x0000 0300 + (0x40 * i)	0x4208 7300 + (0x40 * i)	0x4218 7300 + (0x40 * i)
EDMA_TPTCn_DFSRCi (1)	R	32	0x0000 0304 + (0x40 * i)	0x4208 7304 + (0x40 * i)	0x4218 7304 + (0x40 * i)
EDMA_TPTCn_DFCNTi (1)	R	32	0x0000 0308 + (0x40 * i)	0x4208 7308 + (0x40 * i)	0x4218 7308 + (0x40 * i)
EDMA_TPTCn_DFDSTi (2)	R	32	0x0000 030C + (0x40 * i)	0x4208 730C + (0x40 * i)	0x4218 730C + (0x40 * i)
EDMA_TPTCn_DFBIDXi (2)	R	32	0x0000 0310 + (0x40 * i)	0x4208 7310 + (0x40 * i)	0x4218 7310 + (0x40 * i)
EDMA_TPTCn_DFMPPRXYi (2)	R	32	0x0000 0314 + (0x40 * i)	0x4208 7314 + (0x40 * i)	0x4218 7314 + (0x40 * i)

(1) i = 0 to 1 for EVE1_EDMA_TPTC1 and EVE2_EDMA_TPTC1
(2) i = 0 to 1 for EVE1_EDMA_TPTC1 and EVE2_EDMA_TPTC1

NOTE: EVE is not supported in this family of devices.

Table 16-123. EVE3 and EVE4 EDMA_TPTC0 Registers Mapping Summary (L3_MAIN Access)

Register Name	Type	Register Width (Bits)	Address Offset	EVE3_EDMA_TPTC0 Physical Address (L3_MAIN Access)	EVE4_EDMA_TPTC0 Physical Address (L3_MAIN Access)
EDMA_TPTCn_PID	R	32	0x0000 0000	-	-
EDMA_TPTCn_TCCFG	R	32	0x0000 0004	-	-
EDMA_TPTCn_TCSTAT	R	32	0x0000 0100	0x4228 6100	0x4238 6100
EDMA_TPTCn_INTSTAT	R	32	0x0000 0104	0x4228 6104	0x4238 6104
EDMA_TPTCn_INTEN	RW	32	0x0000 0108	0x4228 6108	0x4238 6108
EDMA_TPTCn_INTCLR	W	32	0x0000 010C	0x4228 610C	0x4238 610C
EDMA_TPTCn_INTCMD	W	32	0x0000 0110	0x4228 6110	0x4238 6110
EDMA_TPTCn_ERRSTAT	R	32	0x0000 0120	0x4228 6120	0x4238 6120
EDMA_TPTCn_ERREN	RW	32	0x0000 0124	0x4228 6124	0x4238 6124

Table 16-123. EVE3 and EVE4 EDMA_TPTC0 Registers Mapping Summary (L3_MAIN Access) (continued)

Register Name	Type	Register Width (Bits)	Address Offset	EVE3_EDMA_TPTC0 Physical Address (L3_MAIN Access)	EVE4_EDMA_TPTC0 Physical Address (L3_MAIN Access)
EDMA_TPTCn_ERRCLR	W	32	0x0000 0128	0x4228 6128	0x4238 6128
EDMA_TPTCn_ERRDET	R	32	0x0000 012C	0x4228 612C	0x4238 612C
EDMA_TPTCn_ERRCMD	W	32	0x0000 0130	0x4228 6130	0x4238 6130
EDMA_TPTCn_RDRATE	RW	32	0x0000 0140	0x4228 6140	0x4238 6140
EDMA_TPTCn_POPT	RW	32	0x0000 0200	0x4228 6200	0x4238 6200
EDMA_TPTCn_PSRC	RW	32	0x0000 0204	0x4228 6204	0x4238 6204
EDMA_TPTCn_PCNT	RW	32	0x0000 0208	0x4228 6208	0x4238 6208
EDMA_TPTCn_PDST	RW	32	0x0000 020C	0x4228 620C	0x4238 620C
EDMA_TPTCn_PBIDX	RW	32	0x0000 0210	0x4228 6210	0x4238 6210
EDMA_TPTCn_PMPPRXY	R	32	0x0000 0214	0x4228 6214	0x4238 6214
EDMA_TPTCn_SAOPT	R	32	0x0000 0240	0x4228 6240	0x4238 6240
EDMA_TPTCn_SASRC	R	32	0x0000 0244	0x4228 6244	0x4238 6244
EDMA_TPTCn_SACNT	R	32	0x0000 0248	0x4228 6248	0x4238 6248
EDMA_TPTCn_SADST	R	32	0x0000 024C	-	-
EDMA_TPTCn_SABIDX	R	32	0x0000 0250	0x4228 6250	0x4238 6250
EDMA_TPTCn_SAMPPRXY	R	32	0x0000 0254	0x4228 6254	0x4238 6254
EDMA_TPTCn_SACNTRLD	R	32	0x0000 0258	0x4228 6258	0x4238 6258
EDMA_TPTCn_SASRCBREF	R	32	0x0000 025C	0x4228 625C	0x4238 625C
EDMA_TPTCn_SADSTBREF	R	32	0x0000 0260	0x4228 6260	0x4238 6260
EDMA_TPTCn_DFCNTRLD	R	32	0x0000 0280	0x4228 6280	0x4238 6280
EDMA_TPTCn_DFSRCBREF	R	32	0x0000 0284	0x4228 6284	0x4238 6284
EDMA_TPTCn_DFDSTBREF	R	32	0x0000 0288	-	-
EDMA_TPTCn_DFOPTi [(1)]	R	32	0x0000 0300 + (0x40 * i)	0x4228 6300 + (0x40 * i)	0x4238 6300 + (0x40 * i)
EDMA_TPTCn_DFSRCi [(1)]	R	32	0x0000 0304 + (0x40 * i)	0x4228 6304 + (0x40 * i)	0x4238 6304 + (0x40 * i)
EDMA_TPTCn_DFCNTi [(1)]	R	32	0x0000 0308 + (0x40 * i)	0x4228 6308 + (0x40 * i)	0x4238 6308 + (0x40 * i)
EDMA_TPTCn_DFDSTi [(1)]	R	32	0x0000 030C + (0x40 * i)	0x4228 630C + (0x40 * i)	0x4238 630C + (0x40 * i)
EDMA_TPTCn_DFBIDXi [(1)]	R	32	0x0000 0310 + (0x40 * i)	0x4228 6310 + (0x40 * i)	0x4238 6310 + (0x40 * i)
EDMA_TPTCn_DFMPPRXYi [(1)]	R	32	0x0000 0314 + (0x40 * i)	0x4228 6314 + (0x40 * i)	0x4238 6314 + (0x40 * i)

(1) i = 0 to 1 for EVE3_EDMA_TPTC0 and EVE4_EDMA_TPTC0

NOTE: EVE is not supported in this family of devices.

Table 16-124. EVE3 and EVE4 EDMA_TPTC1 Registers Mapping Summary (L3_MAIN Access)

Register Name	Type	Register Width (Bits)	Address Offset	EVE3_EDMA_TPTC1 Physical Address (L3_MAIN Access)	EVE4_EDMA_TPTC1 Physical Address (L3_MAIN Access)
EDMA_TPTCn_PID	R	32	0x0000 0000	-	-
EDMA_TPTCn_TCCFG	R	32	0x0000 0004	-	-
EDMA_TPTCn_TCSTAT	R	32	0x0000 0100	0x4228 7100	0x4238 7100
EDMA_TPTCn_INTSTAT	R	32	0x0000 0104	0x4228 7104	0x4238 7104

Table 16-124. EVE3 and EVE4 EDMA_TPTC1 Registers Mapping Summary (L3_MAIN Access) (continued)

Register Name	Type	Register Width (Bits)	Address Offset	EVE3_EDMA_TPTC1 Physical Address (L3_MAIN Access)	EVE4_EDMA_TPTC1 Physical Address (L3_MAIN Access)
EDMA_TPTCn_INTEN	RW	32	0x0000 0108	0x4228 7108	0x4238 7108
EDMA_TPTCn_INTCLR	W	32	0x0000 010C	0x4228 710C	0x4238 710C
EDMA_TPTCn_INTCMD	W	32	0x0000 0110	0x4228 7110	0x4238 7110
EDMA_TPTCn_ERRSTAT	R	32	0x0000 0120	0x4228 7120	0x4238 7120
EDMA_TPTCn_ERREN	RW	32	0x0000 0124	0x4228 7124	0x4238 7124
EDMA_TPTCn_ERRCLR	W	32	0x0000 0128	0x4228 7128	0x4238 7128
EDMA_TPTCn_ERRDET	R	32	0x0000 012C	0x4228 712C	0x4238 712C
EDMA_TPTCn_ERRCMD	W	32	0x0000 0130	0x4228 7130	0x4238 7130
EDMA_TPTCn_RDRATE	RW	32	0x0000 0140	0x4228 7140	0x4238 7140
EDMA_TPTCn_POPT	RW	32	0x0000 0200	0x4228 7200	0x4238 7200
EDMA_TPTCn_PSRC	RW	32	0x0000 0204	0x4228 7204	0x4238 7204
EDMA_TPTCn_PCNT	RW	32	0x0000 0208	0x4228 7208	0x4238 7208
EDMA_TPTCn_PDST	RW	32	0x0000 020C	0x4228 720C	0x4238 720C
EDMA_TPTCn_PBIDX	RW	32	0x0000 0210	0x4228 7210	0x4238 7210
EDMA_TPTCn_PMPPRXY	R	32	0x0000 0214	0x4228 7214	0x4238 7214
EDMA_TPTCn_SAOPT	R	32	0x0000 0240	0x4228 7240	0x4238 7240
EDMA_TPTCn_SASRC	R	32	0x0000 0244	0x4228 7244	0x4238 7244
EDMA_TPTCn_SACNT	R	32	0x0000 0248	0x4228 7248	0x4238 7248
EDMA_TPTCn_SADST	R	32	0x0000 024C	-	-
EDMA_TPTCn_SABIDX	R	32	0x0000 0250	0x4228 7250	0x4238 7250
EDMA_TPTCn_SAMPPRXY	R	32	0x0000 0254	0x4228 7254	0x4238 7254
EDMA_TPTCn_SACNTRLD	R	32	0x0000 0258	0x4228 7258	0x4238 7258
EDMA_TPTCn_SASRCBREF	R	32	0x0000 025C	0x4228 725C	0x4238 725C
EDMA_TPTCn_SADSTBREF	R	32	0x0000 0260	0x4228 7260	0x4238 7260
EDMA_TPTCn_DFCNTRLD	R	32	0x0000 0280	0x4228 7280	0x4238 7280
EDMA_TPTCn_DFSRCBREF	R	32	0x0000 0284	0x4228 7284	0x4238 7284
EDMA_TPTCn_DFDSTBREF	R	32	0x0000 0288	-	-
EDMA_TPTCn_DFOPTi [(1)]	R	32	0x0000 0300 + (0x40 * i)	0x4228 7300 + (0x40 * i)	0x4238 7300 + (0x40 * i)
EDMA_TPTCn_DFSRCi [(1)]	R	32	0x0000 0304 + (0x40 * i)	0x4228 7304 + (0x40 * i)	0x4238 7304 + (0x40 * i)
EDMA_TPTCn_DFCNTi [(1)]	R	32	0x0000 0308 + (0x40 * i)	0x4228 7308 + (0x40 * i)	0x4238 7308 + (0x40 * i)
EDMA_TPTCn_DFDSTi [(1)]	R	32	0x0000 030C + (0x40 * i)	0x4228 730C + (0x40 * i)	0x4238 730C + (0x40 * i)
EDMA_TPTCn_DFBIDXi [(1)]	R	32	0x0000 0310 + (0x40 * i)	0x4228 7310 + (0x40 * i)	0x4238 7310 + (0x40 * i)
EDMA_TPTCn_DFMPPRXYi [(1)]	R	32	0x0000 0314 + (0x40 * i)	0x4228 7314 + (0x40 * i)	0x4238 7314 + (0x40 * i)

(1) i = 0 to 1 for EVE3_EDMA_TPTC1 and EVE4_EDMA_TPTC1

16.2.8.2.2 EDMA Register Description

16.2.8.2.2.1 EDMA_TPCC Register Description

Table 16-125 through Table 16-349 describe the EDMA_TPCC module registers.

Table 16-125. EDMA_TPCC_PID

Address Offset	0x0000 0000		
Physical Address	0x4330 0000 0x40D1 0000 0x4151 0000 0x01D1 0000 0x420A 0000 0x421A 0000 0x422A 0000 0x423A 0000	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Peripheral ID Register		
Type	R		

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

Bits	Field Name	Description	Type	Reset
31:0	REVISION	IP revision	R	TI internal data

Table 16-126. Register Call Summary for Register EDMA_TPCC_PID

Enhanced DMA

- EDMA Register Summary: [0] [1] [2] [3] [4] [5]

Table 16-127. EDMA_TPCC_CCCFG

Address Offset	0x0000 0004		
Physical Address	0x4330 0004 0x40D1 0004 0x4151 0004 0x01D1 0004 0x420A 0004 0x421A 0004 0x422A 0004 0x423A 0004	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	CC Configuration Register		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED						MPEXIST	CHMAPEXIST	RESERVED		NUMREGN		RESERVED	NUMTC			RESERVED	NUMPAENTRY			RESERVED	NUMINTCH			RESERVED	NUMQDMACH			RESERVED	NUMDMACH		

Bits	Field Name	Description	Type	Reset
31:26	RESERVED	Reads return 0's	R	0x0
25	MPEXIST	Memory Protection Existence 0x0: No memory protection 0x1: Memory Protection logic included	R	See Table 16-83
24	CHMAPEXIST	Channel Mapping Existence 0x0: No Channel mapping 0x1: Channel mapping logic included	R	See Table 16-83
23:22	RESERVED	Reads return 0's	R	0x0

Bits	Field Name	Description	Type	Reset
21:20	NUMREGN	Number of MP and Shadow regions 0x0: 0 Regions 0x1: 2 Regions 0x2: 4 Regions 0x3: 8 Regions	R	See Table 16-83
19	RESERVED	Reads return 0's	R	0x0
18:16	NUMTC	Number of Queues/Number of TCs 0x0: 1 TC/Event Queue 0x1: 2 TC/Event Queue 0x2: 3 TC/Event Queue 0x3: 4 TC/Event Queue 0x4: 5 TC/Event Queue 0x5: 6 TC/Event Queue 0x6: 7 TC/Event Queue 0x7: 8 TC/Event Queue	R	See Table 16-83
15	RESERVED	Reads return 0's	R	0x0
14:12	NUMPAENTRY	Number of PaRAM entries 0x0: 16 entries 0x1: 32 entries 0x2: 64 entries 0x3: 128 entries 0x4: 256 entries 0x5: 512 entries	R	See Table 16-83
11	RESERVED	Reads return 0's	R	0x0
10:8	NUMINTCH	Number of Interrupt Channels 0x1: 8 Interrupt channels 0x2: 16 Interrupt channels 0x3: 32 Interrupt channels 0x4: 64 Interrupt channels	R	See Table 16-83
7	RESERVED	reads return 0's	R	0x0
6:4	NUMQDMACH	Number of QDMA Channels 0x0: No QDMA Channels 0x1: 2 QDMA Channels 0x2: 4 QDMA Channels 0x3: 6 QDMA Channels 0x4: 8 QDMA Channels	R	See Table 16-83
3	RESERVED	reads return 0's	R	0x0
2:0	NUMDMACH	Number of DMA Channels 0x0: No DMA Channels 0x1: 4 DMA Channels 0x2: 8 DMA Channels 0x3: 16 DMA Channels 0x4: 32 DMA Channels 0x5: 64 DMA Channels	R	See Table 16-83

Table 16-128. Register Call Summary for Register EDMA_TPCC_CCCFG

Enhanced DMA

- EDMA Register Summary: [0] [1] [2] [3] [4] [5]

Table 16-129. EDMA_TPCC_CLKGDIS

Address Offset	0x0000 00FC		
Physical Address	0x4330 00FC 0x40D1 00FC 0x4151 00FC 0x01D1 00FC 0x420A 00FC 0x421A 00FC 0x422A 00FC 0x423A 00FC	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Auto Clock Gate Disable		
Type	RW		

31–1	0
RESERVED	CLKGDIS

Bits	Field Name	Description	Type	Reset
31:1	RESERVED	Reserved	R	0x0
0	CLKGDIS	Auto Clock Gate Disable	RW	0x0

Table 16-130. Register Call Summary for Register EDMA_TPCC_CLKGDIS

Enhanced DMA
- Clock and Power Management: [0]
- EDMA Register Summary: [1] [2] [3] [4] [5] [6]

Table 16-131. EDMA_TPCC_DCHMAPN_m

Address Offset	0x0000 0100 + (0x4 * m)		
Physical Address	0x4330 0100 + (0x4 * m) 0x40D1 0100 + (0x4 * m) 0x4151 0100 + (0x4 * m) 0x01D1 0100 + (0x4 * m) 0x420A 0100 + (0x4 * m) 0x421A 0100 + (0x4 * m) 0x422A 0100 + (0x4 * m) 0x423A 0100 + (0x4 * m)	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	DMA Channel N Mapping Register		
Type	RW		

31–14	13–5	4–0
RESERVED	PAENTRY	RESERVED

Bits	Field Name	Description	Type	Reset
31:14	RESERVED	Reserved	R	0x0
13:5	PAENTRY	PaRAM Entry number for DMA Channel N.	RW	0x0
4:0	RESERVED	Reserved	R	0x0

Table 16-132. Register Call Summary for Register EDMA_TPCC_DCHMAPN_m

Enhanced DMA

- Parameter RAM (PaRAM): [0]
- DMA Channel to PaRAM Mapping: [1] [2] [3]
- Setting Up an EDMA Transfer: [4]
- EDMA Register Summary: [5] [6] [7] [8] [9] [10]

Table 16-133. EDMA_TPCC_QCHMAPN_j

Address Offset	0x0000 0200 + (0x4 * j)		
Physical Address	0x4330 0200 + (0x4 * j) 0x40D1 0200 + (0x4 * j) 0x4151 0200 + (0x4 * j) 0x01D1 0200 + (0x4 * j) 0x420A 0200 + (0x4 * j) 0x421A 0200 + (0x4 * j) 0x422A 0200 + (0x4 * j) 0x423A 0200 + (0x4 * j)	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	QDMA Channel N Mapping Register		
Type	RW		

31:14	13:5	4:2	1:0
RESERVED	PAENTRY	TRWORD	RESERVED

Bits	Field Name	Description	Type	Reset
31:14	RESERVED	Reserved	R	0x0
13:5	PAENTRY	PaRAM Entry number for QDMA Channel N.	RW	0x0
4:2	TRWORD	TRWORD points to the specific trigger word of the PaRAM Entry defined by PAENTRY. A write to the trigger word results in a QDMA Event being recognized.	RW	0x0
1:0	RESERVED	Reserved	R	0x0

Table 16-134. Register Call Summary for Register EDMA_TPCC_QCHMAPN_j

Enhanced DMA

- Parameter RAM (PaRAM): [0]
- Linking Transfers: [1]
- QDMA Channels: [2] [3]
- QDMA Channel to PaRAM Mapping: [4] [5] [6] [7]
- Setting Up an EDMA Transfer: [8]
- EDMA Register Summary: [9] [10] [11] [12] [13] [14]

Table 16-135. EDMA_TPCC_DMAQNUMN_k

Address Offset	0x0000 0240 + (0x4 * k)		
Physical Address	0x4330 0240 + (0x4 * k) 0x40D1 0240 + (0x4 * k) 0x4151 0240 + (0x4 * k) 0x01D1 0240 + (0x4 * k) 0x420A 0240 + (0x4 * k) 0x421A 0240 + (0x4 * k) 0x422A 0240 + (0x4 * k) 0x423A 0240 + (0x4 * k)	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC

Table 16-135. EDMA_TPCC_DMAQNUMN_k (continued)

Description	DMA Queue Number Register n Contains the Event queue number to be used for the corresponding DMA Channel.
Type	RW

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED	E7			RESERVED	E6			RESERVED	E5			RESERVED	E4			RESERVED	E3			RESERVED	E2			RESERVED	E1			RESERVED	E0		

Bits	Field Name	Description	Type	Reset
31	RESERVED	Reserved	R	0x0
30:28	E7	DMA Queue Number for event #7	RW	0x0
27	RESERVED	Reserved	R	0x0
26:24	E6	DMA Queue Number for event #6	RW	0x0
23	RESERVED	Reserved	R	0x0
22:20	E5	DMA Queue Number for event #5	RW	0x0
19	RESERVED	Reserved	R	0x0
18:16	E4	DMA Queue Number for event #4	RW	0x0
15	RESERVED	Reserved	R	0x0
14:12	E3	DMA Queue Number for event #3	RW	0x0
11	RESERVED	Reserved	R	0x0
10:8	E2	DMA Queue Number for event #2	RW	0x0
7	RESERVED	Reserved	R	0x0
6:4	E1	DMA Queue Number for event #1	RW	0x0
3	RESERVED	Reserved	R	0x0
2:0	E0	DMA Queue Number for event #0	RW	0x0

Table 16-136. Register Call Summary for Register EDMA_TPCC_DMAQNUMN_k

Enhanced DMA

- DMA/QDMA Channel to Event Queue Mapping: [0]
- Dequeue Priority: [1]
- Setting Up an EDMA Transfer: [2]
- EDMA Register Summary: [3] [4] [5] [6] [7] [8]

Table 16-137. EDMA_TPCC_QDMAQNUM

Address Offset	0x0000 0260		
Physical Address	0x4330 0260 0x40D1 0260 0x4151 0260 0x01D1 0260 0x420A 0260 0x421A 0260 0x422A 0260 0x423A 0260	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	QDMA Queue Number Register Contains the Event queue number to be used for the corresponding QDMA Channel.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED	E7			RESERVED	E6			RESERVED	E5			RESERVED	E4			RESERVED	E3			RESERVED	E2			RESERVED	E1			RESERVED	E0		

Bits	Field Name	Description	Type	Reset
31	RESERVED		R	0x0
30:28	E7	QDMA Queue Number for event #7	RW	0x0
27	RESERVED		R	0x0
26:24	E6	QDMA Queue Number for event #6	RW	0x0
23	RESERVED		R	0x0
22:20	E5	QDMA Queue Number for event #5	RW	0x0
19	RESERVED		R	0x0
18:16	E4	QDMA Queue Number for event #4	RW	0x0
15	RESERVED		R	0x0
14:12	E3	QDMA Queue Number for event #3	RW	0x0
11	RESERVED		R	0x0
10:8	E2	QDMA Queue Number for event #2	RW	0x0
7	RESERVED		R	0x0
6:4	E1	QDMA Queue Number for event #1	RW	0x0
3	RESERVED		R	0x0
2:0	E0	QDMA Queue Number for event #0	RW	0x0

Table 16-138. Register Call Summary for Register EDMA_TPCC_QDMAQNUM

Enhanced DMA

- DMA/QDMA Channel to Event Queue Mapping: [0]
- Dequeue Priority: [1]
- Setting Up an EDMA Transfer: [2]
- EDMA Register Summary: [3] [4] [5] [6] [7] [8]

Table 16-139. EDMA_TPCC_QUETCMAP

Address Offset	0x0000 0280		
Physical Address	0x4330 0280 0x40D1 0280 0x4151 0280 0x01D1 0280 0x420A 0280 0x421A 0280 0x422A 0280 0x423A 0280	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Queue to TC Mapping		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									TCNUMQ1			RESERVED	TCNUMQ0		

Bits	Field Name	Description	Type	Reset
31:7	RESERVED	Reserved	R	0x0
6:4	TCNUMQ1	TC Number for Queue N: Defines the TC number that Event Queue N TRs are written to.	RW	0x1
3	RESERVED	Reserved	R	0x0
2:0	TCNUMQ0	TC Number for Queue N: Defines the TC number that Event Queue N TRs are written to.	RW	0x0

Table 16-140. Register Call Summary for Register EDMA_TPCC_QUETCMAP

Enhanced DMA

- EDMA Register Summary: [0] [1] [2] [3] [4] [5]

Table 16-141. EDMA_TPCC_QUEPRI

Address Offset	0x0000 0284		
Physical Address	0x4330 0284 0x40D1 0284 0x4151 0284 0x01D1 0284 0x420A 0284 0x421A 0284 0x422A 0284 0x423A 0284	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Queue Priority		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									PRIQ1			RESERVED	PRIQ0		

Bits	Field Name	Description	Type	Reset
31:7	RESERVED	Reserved	R	0x0
6:4	PRIQ1	Priority Level for Queue 1 Dictates the priority level used for the OPTIONS field programmation for Qn TRs. Sets the priority used for TC read and write commands.	RW	0x0
3	RESERVED	Reserved	R	0x0
2:0	PRIQ0	Priority Level for Queue 0 Dictates the priority level used for the OPTIONS field programmation for Qn TRs. Sets the priority used for TC read and write commands.	RW	0x0

Table 16-142. Register Call Summary for Register EDMA_TPCC_QUEPRI

Enhanced DMA

- Performance Considerations: [0]
- EDMA Register Summary: [1] [2] [3] [4] [5] [6]

Table 16-143. EDMA_TPCC_EMR

Address Offset	0x0000 0300		
Physical Address	0x4330 0300 0x40D1 0300 0x4151 0300 0x01D1 0300 0x420A 0300 0x421A 0300 0x422A 0300 0x423A 0300	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Event Missed Register: The Event Missed register is set if 2 events are received without the first event being cleared or if a Null TR is serviced. Chained events (CER), Set Events (ESR), and normal events (ER) are treated individually. If any bit in the EMR register is set (and all errors (including EDMA_TPCC_QEMR / EDMA_TPCC_CCERR) were previously clear), then an error will be signaled with TPCC error interrupt.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E31	E30	E29	E28	E27	E26	E25	E24	E23	E22	E21	E20	E19	E18	E17	E16	E15	E14	E13	E12	E11	E10	E9	E8	E7	E6	E5	E4	E3	E2	E1	E0

Bits	Field Name	Description	Type	Reset
31	E31	Event Missed #31	R	0x0
30	E30	Event Missed #30	R	0x0
29	E29	Event Missed #29	R	0x0
28	E28	Event Missed #28	R	0x0
27	E27	Event Missed #27	R	0x0
26	E26	Event Missed #26	R	0x0
25	E25	Event Missed #25	R	0x0
24	E24	Event Missed #24	R	0x0
23	E23	Event Missed #23	R	0x0
22	E22	Event Missed #22	R	0x0
21	E21	Event Missed #21	R	0x0
20	E20	Event Missed #20	R	0x0
19	E19	Event Missed #19	R	0x0
18	E18	Event Missed #18	R	0x0
17	E17	Event Missed #17	R	0x0
16	E16	Event Missed #16	R	0x0
15	E15	Event Missed #15	R	0x0
14	E14	Event Missed #14	R	0x0
13	E13	Event Missed #13	R	0x0
12	E12	Event Missed #12	R	0x0
11	E11	Event Missed #11	R	0x0
10	E10	Event Missed #10	R	0x0
9	E9	Event Missed #9	R	0x0
8	E8	Event Missed #8	R	0x0
7	E7	Event Missed #7	R	0x0
6	E6	Event Missed #6	R	0x0
5	E5	Event Missed #5	R	0x0
4	E4	Event Missed #4	R	0x0
3	E3	Event Missed #3	R	0x0
2	E2	Event Missed #2	R	0x0
1	E1	Event Missed #1	R	0x0

Bits	Field Name	Description	Type	Reset
0	E0	Event Missed #0	R	0x0

Table 16-144. Register Call Summary for Register EDMA_TPCC_EMR

Enhanced DMA

- Third-Party Channel Controller: [0]
- Null PaRAM Set: [1]
- Dummy PaRAM Set: [2]
- Dummy Versus Null Transfer Comparison: [3] [4]
- DMA Channel: [5] [6] [7] [8] [9] [10]
- Interrupt Evaluation Operations: [11]
- Error Interrupts: [12]
- EDMA Debug Checklist: [13] [14]
- EDMA Register Summary: [15] [16] [17] [18] [19] [20]
- EDMA Register Description: [21] [22] [23] [24] [25] [26] [27] [28] [29] [30] [31] [32] [33]

Table 16-145. EDMA_TPCC_EMRH

Address Offset	0x0000 0304		
Physical Address	0x4330 0304 0x40D1 0304 0x4151 0304 0x01D1 0304 0x420A 0304 0x421A 0304 0x422A 0304 0x423A 0304	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Event Missed Register (High Part): The Event Missed register is set if 2 events are received without the first event being cleared or if a Null TR is serviced. Chained events (CER), Set Events (ESR), and normal events (ER) are treated individually. If any bit in the EMR register is set (and all errors (including EDMA_TPCC_QEMR / EDMA_TPCC_CCERR) were previously clear), then an error will be signaled with TPCC error interrupt.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E63	E62	E61	E60	E59	E58	E57	E56	E55	E54	E53	E52	E51	E50	E49	E48	E47	E46	E45	E44	E43	E42	E41	E40	E39	E38	E37	E36	E35	E34	E33	E32

Bits	Field Name	Description	Type	Reset
31	E63	Event Missed #63	R	0x0
30	E62	Event Missed #62	R	0x0
29	E61	Event Missed #61	R	0x0
28	E60	Event Missed #60	R	0x0
27	E59	Event Missed #59	R	0x0
26	E58	Event Missed #58	R	0x0
25	E57	Event Missed #57	R	0x0
24	E56	Event Missed #56	R	0x0
23	E55	Event Missed #55	R	0x0
22	E54	Event Missed #54	R	0x0
21	E53	Event Missed #53	R	0x0
20	E52	Event Missed #52	R	0x0
19	E51	Event Missed #51	R	0x0
18	E50	Event Missed #50	R	0x0
17	E49	Event Missed #49	R	0x0
16	E48	Event Missed #48	R	0x0

Bits	Field Name	Description	Type	Reset
15	E47	Event Missed #47	R	0x0
14	E46	Event Missed #46	R	0x0
13	E45	Event Missed #45	R	0x0
12	E44	Event Missed #44	R	0x0
11	E43	Event Missed #43	R	0x0
10	E42	Event Missed #42	R	0x0
9	E41	Event Missed #41	R	0x0
8	E40	Event Missed #40	R	0x0
7	E39	Event Missed #39	R	0x0
6	E38	Event Missed #38	R	0x0
5	E37	Event Missed #37	R	0x0
4	E36	Event Missed #36	R	0x0
3	E35	Event Missed #35	R	0x0
2	E34	Event Missed #34	R	0x0
1	E33	Event Missed #33	R	0x0
0	E32	Event Missed #32	R	0x0

Table 16-146. Register Call Summary for Register EDMA_TPCC_EMRH

Enhanced DMA

- Third-Party Channel Controller: [0]
- Null PaRAM Set: [1]
- Dummy PaRAM Set: [2]
- Dummy Versus Null Transfer Comparison: [3]
- Interrupt Evaluation Operations: [4]
- Error Interrupts: [5]
- EDMA Debug Checklist: [6]
- EDMA Register Summary: [7] [8] [9] [10] [11] [12]
- EDMA Register Description: [13] [14]

Table 16-147. EDMA_TPCC_EMCR

Address Offset	0x0000 0308		
Physical Address	0x4330 0308 0x40D1 0308 0x4151 0308 0x01D1 0308 0x420A 0308 0x421A 0308 0x422A 0308 0x423A 0308	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Event Missed Clear Register: CPU write of '1' to the EDMA_TPCC_EMCR.En bit causes the EDMA_TPCC_EMR.En bit to be cleared. CPU write of '0' has no effect.. All error bits must be cleared before additional error interrupts will be asserted by CC.		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E31	E30	E29	E28	E27	E26	E25	E24	E23	E22	E21	E20	E19	E18	E17	E16	E15	E14	E13	E12	E11	E10	E9	E8	E7	E6	E5	E4	E3	E2	E1	E0

Bits	Field Name	Description	Type	Reset
31	E31	Event Missed Clear #31	W	0x0
30	E30	Event Missed Clear #30	W	0x0
29	E29	Event Missed Clear #29	W	0x0

Bits	Field Name	Description	Type	Reset
28	E28	Event Missed Clear #28	W	0x0
27	E27	Event Missed Clear #27	W	0x0
26	E26	Event Missed Clear #26	W	0x0
25	E25	Event Missed Clear #25	W	0x0
24	E24	Event Missed Clear #24	W	0x0
23	E23	Event Missed Clear #23	W	0x0
22	E22	Event Missed Clear #22	W	0x0
21	E21	Event Missed Clear #21	W	0x0
20	E20	Event Missed Clear #20	W	0x0
19	E19	Event Missed Clear #19	W	0x0
18	E18	Event Missed Clear #18	W	0x0
17	E17	Event Missed Clear #17	W	0x0
16	E16	Event Missed Clear #16	W	0x0
15	E15	Event Missed Clear #15	W	0x0
14	E14	Event Missed Clear #14	W	0x0
13	E13	Event Missed Clear #13	W	0x0
12	E12	Event Missed Clear #12	W	0x0
11	E11	Event Missed Clear #11	W	0x0
10	E10	Event Missed Clear #10	W	0x0
9	E9	Event Missed Clear #9	W	0x0
8	E8	Event Missed Clear #8	W	0x0
7	E7	Event Missed Clear #7	W	0x0
6	E6	Event Missed Clear #6	W	0x0
5	E5	Event Missed Clear #5	W	0x0
4	E4	Event Missed Clear #4	W	0x0
3	E3	Event Missed Clear #3	W	0x0
2	E2	Event Missed Clear #2	W	0x0
1	E1	Event Missed Clear #1	W	0x0
0	E0	Event Missed Clear #0	W	0x0

Table 16-148. Register Call Summary for Register EDMA_TPCC_EMCR

Enhanced DMA

- EDMA Register Summary: [0] [1] [2] [3] [4] [5]
- EDMA Register Description: [6] [7]

Table 16-149. EDMA_TPCC_EMCRH

Address Offset	0x0000 030C		
Physical Address	0x4330 030C 0x40D1 030C 0x4151 030C 0x01D1 030C 0x420A 030C 0x421A 030C 0x422A 030C 0x423A 030C	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Event Missed Clear Register (High Part): CPU write of '1' to the EDMA_TPCC_EMCR.En bit causes the EDMA_TPCC_EMR.En bit to be cleared. CPU write of '0' has no effect. All error bits must be cleared before additional error interrupts will be asserted by CC.		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E63	E62	E61	E60	E59	E58	E57	E56	E55	E54	E53	E52	E51	E50	E49	E48	E47	E46	E45	E44	E43	E42	E41	E40	E39	E38	E37	E36	E35	E34	E33	E32

Bits	Field Name	Description	Type	Reset
31	E63	Event Missed Clear #63	W	0x0
30	E62	Event Missed Clear #62	W	0x0
29	E61	Event Missed Clear #61	W	0x0
28	E60	Event Missed Clear #60	W	0x0
27	E59	Event Missed Clear #59	W	0x0
26	E58	Event Missed Clear #58	W	0x0
25	E57	Event Missed Clear #57	W	0x0
24	E56	Event Missed Clear #56	W	0x0
23	E55	Event Missed Clear #55	W	0x0
22	E54	Event Missed Clear #54	W	0x0
21	E53	Event Missed Clear #53	W	0x0
20	E52	Event Missed Clear #52	W	0x0
19	E51	Event Missed Clear #51	W	0x0
18	E50	Event Missed Clear #50	W	0x0
17	E49	Event Missed Clear #49	W	0x0
16	E48	Event Missed Clear #48	W	0x0
15	E47	Event Missed Clear #47	W	0x0
14	E46	Event Missed Clear #46	W	0x0
13	E45	Event Missed Clear #45	W	0x0
12	E44	Event Missed Clear #44	W	0x0
11	E43	Event Missed Clear #43	W	0x0
10	E42	Event Missed Clear #42	W	0x0
9	E41	Event Missed Clear #41	W	0x0
8	E40	Event Missed Clear #40	W	0x0
7	E39	Event Missed Clear #39	W	0x0
6	E38	Event Missed Clear #38	W	0x0
5	E37	Event Missed Clear #37	W	0x0
4	E36	Event Missed Clear #36	W	0x0
3	E35	Event Missed Clear #35	W	0x0
2	E34	Event Missed Clear #34	W	0x0
1	E33	Event Missed Clear #33	W	0x0
0	E32	Event Missed Clear #32	W	0x0

Table 16-150. Register Call Summary for Register EDMA_TPCC_EMCRH

Enhanced DMA
- EDMA Register Summary: [0] [1] [2] [3] [4] [5]

Table 16-151. EDMA_TPCC_QEMR

Address Offset	0x0000 0310		
Physical Address	0x4330 0310 0x40D1 0310 0x4151 0310 0x01D1 0310 0x420A 0310 0x421A 0310 0x422A 0310 0x423A 0310	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC

Table 16-151. EDMA_TPCC_QEMR (continued)

Description	QDMA Event Missed Register: The QDMA Event Missed register is set if 2 QDMA events are detected without the first event being cleared or if a Null TR is serviced. If any bit in the EDMA_TPCC_QEMR register is set (and all errors (including EDMA_TPCC_EMR / EDMA_TPCC_CCERR) were previously clear), then an error will be signaled with TPCC error interrupt.
Type	R

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	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

Bits	Field Name	Description	Type	Reset
31:8	RESERVED	Reserved	R	0x0
7	E7	Event Missed #7	R	0x0
6	E6	Event Missed #6	R	0x0
5	E5	Event Missed #5	R	0x0
4	E4	Event Missed #4	R	0x0
3	E3	Event Missed #3	R	0x0
2	E2	Event Missed #2	R	0x0
1	E1	Event Missed #1	R	0x0
0	E0	Event Missed #0	R	0x0

Table 16-152. Register Call Summary for Register EDMA_TPCC_QEMR

Enhanced DMA

- Null PaRAM Set: [0]
- Dummy PaRAM Set: [1]
- Dummy Versus Null Transfer Comparison: [2]
- QDMA Channels: [3]
- Interrupt Evaluation Operations: [4]
- Error Interrupts: [5]
- EDMA Debug Checklist: [6] [7]
- EDMA Register Summary: [8] [9] [10] [11] [12] [13]
- EDMA Register Description: [14] [15] [16] [17] [18] [19] [20] [21] [22] [23] [24] [25] [26] [27]

Table 16-153. EDMA_TPCC_QEMCR

Address Offset	0x0000 0314		
Physical Address	0x4330 0314 0x40D1 0314 0x4151 0314 0x01D1 0314 0x420A 0314 0x421A 0314 0x422A 0314 0x423A 0314	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	QDMA Event Missed Clear Register: CPU write of '1' to the EDMA_TPCC_QEMCR.En bit causes the EDMA_TPCC_QEMR.En bit to be cleared. CPU write of '0' has no effect. All error bits must be cleared before additional error interrupts will be asserted by CC.		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	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

Bits	Field Name	Description	Type	Reset
31:8	RESERVED	Reserved	R	0x0
7	E7	Event Missed Clear #7	W	0x0
6	E6	Event Missed Clear #6	W	0x0
5	E5	Event Missed Clear #5	W	0x0
4	E4	Event Missed Clear #4	W	0x0
3	E3	Event Missed Clear #3	W	0x0
2	E2	Event Missed Clear #2	W	0x0
1	E1	Event Missed Clear #1	W	0x0
0	E0	Event Missed Clear #0	W	0x0

Table 16-154. Register Call Summary for Register EDMA_TPCC_QEMCR

Enhanced DMA

- EDMA Register Summary: [0] [1] [2] [3] [4] [5]
- EDMA Register Description: [6]

Table 16-155. EDMA_TPCC_CCERR

Address Offset	0x0000 0318		
Physical Address	0x4330 0318 0x40D1 0318 0x4151 0318 0x01D1 0318 0x420A 0318 0x421A 0318 0x422A 0318 0x423A 0318	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	CC Error Register		
Type	R		

31:17	16	15:8	7	6	5	4	3	2	1	0
RESERVED	TCERR	RESERVED	QTHRXCD7	QTHRXCD6	QTHRXCD5	QTHRXCD4	QTHRXCD3	QTHRXCD2	QTHRXCD1	QTHRXCD0

Bits	Field Name	Description	Type	Reset
31:17	RESERVED	Reserved	R	0x0
16	TCERR	Transfer Completion Code Error 0x0: Total number of allowed TCCs outstanding has not been reached. 0x1: Total number of allowed TCCs has been reached. TCERR can be cleared by writing a '1' to corresponding bit in EDMA_TPCC_CCERRCLR register. If any bit in the EDMA_TPCC_CCERR register is set (and all errors were previously clear), then an error will be signaled with TPCC error interrupt.	R	0x0
15:8	RESERVED	Reserved	R	0x0

Bits	Field Name	Description	Type	Reset
7	QTHRXCD7	Queue Threshold Error for Q7 0x0: Watermark/threshold has not been exceeded. 0x1: Watermark/threshold has been exceeded. QTHRXCD7 can be cleared by writing a '1' to corresponding bit in EDMA_TPCC_CCERRCLR register. If any bit in the EDMA_TPCC_CCERR register is set (and all errors (including EDMA_TPCC_EMR/EDMA_TPCC_QEMR) were previously clear), then an error will be signaled with the TPCC error interrupt.	R	0x0
6	QTHRXCD6	Queue Threshold Error for Q6 0x0 : Watermark/threshold has not been exceeded. 0x1 : Watermark/threshold has been exceeded. QTHRXCD6 can be cleared by writing a '1' to corresponding bit in EDMA_TPCC_CCERRCLR register. If any bit in the EDMA_TPCC_CCERR register is set (and all errors (including EDMA_TPCC_EMR/EDMA_TPCC_QEMR) were previously clear), then an error will be signaled with the TPCC error interrupt.	R	0x0
5	QTHRXCD5	Queue Threshold Error for Q5 0x0 : Watermark/threshold has not been exceeded. 0x1 : Watermark/threshold has been exceeded. QTHRXCD5 can be cleared by writing a '1' to corresponding bit in EDMA_TPCC_CCERRCLR register. If any bit in the EDMA_TPCC_CCERR register is set (and all errors (including EDMA_TPCC_EMR/EDMA_TPCC_QEMR) were previously clear), then an error will be signaled with the TPCC error interrupt.	R	0x0
4	QTHRXCD4	Queue Threshold Error for Q4 0x0: Watermark/threshold has not been exceeded. 0x1: Watermark/threshold has been exceeded. QTHRXCD4 can be cleared by writing a '1' to corresponding bit in EDMA_TPCC_CCERRCLR register. If any bit in the EDMA_TPCC_CCERR register is set (and all errors (including EDMA_TPCC_EMR/EDMA_TPCC_QEMR) were previously clear), then an error will be signaled with the TPCC error interrupt.	R	0x0
3	QTHRXCD3	Queue Threshold Error for Q3 0x0: Watermark/threshold has not been exceeded. 0x1 : Watermark/threshold has been exceeded. QTHRXCD3 can be cleared by writing a '1' to corresponding bit in EDMA_TPCC_CCERRCLR register. If any bit in the EDMA_TPCC_CCERR register is set (and all errors (including EDMA_TPCC_EMR/EDMA_TPCC_QEMR) were previously clear), then an error will be signaled with the TPCC error interrupt.	R	0x0
2	QTHRXCD2	Queue Threshold Error for Q2 0x0: Watermark/threshold has not been exceeded. 0x1: Watermark/threshold has been exceeded. QTHRXCD2 can be cleared by writing a '1' to corresponding bit in EDMA_TPCC_CCERRCLR register. If any bit in the EDMA_TPCC_CCERR register is set (and all errors (including EDMA_TPCC_EMR/EDMA_TPCC_QEMR) were previously clear), then an error will be signaled with the TPCC error interrupt.	R	0x0

Bits	Field Name	Description	Type	Reset
1	QTHRXCD1	Queue Threshold Error for Q1 0x0: Watermark/threshold has not been exceeded. 0x1: Watermark/threshold has been exceeded. QTHRXCD1 can be cleared by writing a '1' to corresponding bit in EDMA_TPCC_CCERRCLR register. If any bit in the EDMA_TPCC_CCERR register is set (and all errors (including EDMA_TPCC_EMR/EDMA_TPCC_QEMR) were previously clear), then an error will be signaled with the TPCC error interrupt.	R	0x0
0	QTHRXCD0	Queue Threshold Error for Q0: 0x0: Watermark/threshold has not been exceeded. 0x1: Watermark/threshold has been exceeded. QTHRXCD0 can be cleared by writing a '1' to corresponding bit in EDMA_TPCC_CCERRCLR register. If any bit in the EDMA_TPCC_CCERR register is set (and all errors (including EDMA_TPCC_EMR/EDMA_TPCC_QEMR) were previously clear), then an error will be signaled with the TPCC error interrupt.	R	0x0

Table 16-156. Register Call Summary for Register EDMA_TPCC_CCERR

Enhanced DMA
- Third-Party Channel Controller: [0]
- Interrupt Evaluation Operations: [1]
- Error Interrupts: [2] [3]
- Queue Resource Tracking: [4] [5]
- EDMA Register Summary: [6] [7] [8] [9] [10] [11]
- EDMA Register Description: [12] [13] [14] [15] [16] [17] [18] [19] [20] [21] [22] [23] [24] [25] [26] [27] [28] [29] [30] [31] [32] [33] [34] [35] [36] [37] [38] [39] [40] [41] [42] [43] [44] [45] [46] [47]

Table 16-157. EDMA_TPCC_CCERRCLR

Address Offset	0x0000 031C		
Physical Address	0x4330 031C 0x40D1 031C 0x4151 031C 0x01D1 031C 0x420A 031C 0x421A 031C 0x422A 031C 0x423A 031C	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	CC Error Clear Register		
Type	W		

31–17	16	15–8	7	6	5	4	3	2	1	0
RESERVED	TCERR	RESERVED	QTHRXCD7	QTHRXCD6	QTHRXCD5	QTHRXCD4	QTHRXCD3	QTHRXCD2	QTHRXCD1	QTHRXCD0

Bits	Field Name	Description	Type	Reset
31:17	RESERVED	Reserved	R	0x0
16	TCERR	Clear Error for EDMA_TPCC_CCERR[16] TR. Write 0x1 to clear the value of EDMA_TPCC_CCERR[16] TCERR. Write 0x0 have no affect.	W	0x0
15:8	RESERVED	Reserved	R	0x0
7	QTHRXCD7	Clear error for EDMA_TPCC_CCERR[7]QTHRXCD7 Write 0x0 have no affect. Write 0x1 to clear the values of QSTAT7.WM, QSTAT7.THRXCD, EDMA_TPCC_CCERR[7] QTHRXCD7	W	0x0
6	QTHRXCD6	Clear error for EDMA_TPCC_CCERR[6] QTHRXCD6 Write 0x0 have no affect. Write 0x1 to clear the values of QSTAT6.WM, QSTAT6.THRXCD, EDMA_TPCC_CCERR[6]QTHRXCD6	W	0x0
5	QTHRXCD5	Clear error for EDMA_TPCC_CCERR[5] QTHRXCD5 Write 0x0 have no affect. Write 0x1 to clear the values of QSTAT5.WM, QSTAT5.THRXCD, EDMA_TPCC_CCERR[5]QTHRXCD5	W	0x0
4	QTHRXCD4	Clear error for EDMA_TPCC_CCERR[4] QTHRXCD4: Write 0x0 have no affect. Write 0x1 to clear the values of QSTAT4.WM, QSTAT4.THRXCD, EDMA_TPCC_CCERR[4] QTHRXCD4	W	0x0
3	QTHRXCD3	Clear error for EDMA_TPCC_CCERR[3] QTHRXCD3 Write 0x1 to clear the values of QSTAT3.WM, QSTAT3.THRXCD, EDMA_TPCC_CCERR[3] QTHRXCD3 Write 0x0 have no affect.	W	0x0
2	QTHRXCD2	Clear error for EDMA_TPCC_CCERR[2] QTHRXCD2 Write 0x0 have no affect. Write 0x1 to clear the values of QSTAT2.WM, QSTAT2.THRXCD, EDMA_TPCC_CCERR[2] QTHRXCD2	W	0x0
1	QTHRXCD1	Clear error for EDMA_TPCC_CCERR[1] QTHRXCD1 Write 0x1 to clear the values of QSTAT1.WM, QSTAT1.THRXCD, EDMA_TPCC_CCERR[1] QTHRXCD1 Write 0x0 have no affect.	W	0x0
0	QTHRXCD0	Clear error for EDMA_TPCC_CCERR[0] QTHRXCD0 Write 0x0 have no affect. Write 0x1 to clear the values of QSTAT0.WM, QSTAT0.THRXCD, EDMA_TPCC_CCERR[0] QTHRXCD0	W	0x0

Table 16-158. Register Call Summary for Register EDMA_TPCC_CCERRCLR

Enhanced DMA
- EDMA Register Summary: [0] [1] [2] [3] [4] [5]
- EDMA Register Description: [6] [7] [8] [9] [10] [11] [12] [13] [14]

Table 16-159. EDMA_TPCC_EEVAL

Address Offset	0x0000 0320		
Physical Address	0x4330 0320 0x40D1 0320 0x4151 0320 0x01D1 0320 0x420A 0320 0x421A 0320 0x422A 0320 0x423A 0320	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Error Eval Register		
Type	W		

31:2	1	0
RESERVED	SET	EVAL

Bits	Field Name	Description	Type	Reset
31:2	RESERVED	Reserved	R	0x000000
1	SET	Error Interrupt Set CPU writes 0x0 has no effect. CPU writes 0x1 to the SET bit causes the TPCC error interrupt to be pulsed regardless of state of EDMA_TPCC_EMR/EDMA_TPCC_EMRH, EDMA_TPCC_QEMR, or EDMA_TPCC_CCERR.	W	0x0
0	EVAL	Error Interrupt Evaluate CPU writes 0x0 has no effect. CPU writes 0x1 to the EVAL bit causes the TPCC error interrupt to be pulsed if any errors have not been cleared in the EDMA_TPCC_EMR/EDMA_TPCC_EMRH, EDMA_TPCC_QEMR, or EDMA_TPCC_CCERR registers. The CPU must also write 0x1 after any error interrupts are serviced in order for subsequent interrupts to be asserted.	W	0x0

Table 16-160. Register Call Summary for Register EDMA_TPCC_EEVAL

Enhanced DMA

- Interrupt Evaluation Operations: [0] [1]
- Error Interrupts: [2]
- EDMA Register Summary: [3] [4] [5] [6] [7] [8]

Table 16-161. EDMA_TPCC_DRAEM_k

Address Offset	0x0000 0340 + (0x8 * k)		
Physical Address	0x4330 0340 + (0x8 * k) 0x40D1 0340 + (0x8 * k) 0x4151 0340 + (0x8 * k) 0x01D1 0340 + (0x8 * k)	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC
Description	DMA Region Access enable for bit N in Region M: En = 0 : Accesses via Region M address space to Bit N in any DMA Channel Register are not allowed. Reads will return 'b0 on Bit N and writes will not modify the state of bit N. Enabled interrupt bits for bit N do not contribute to the generation of the TPCC region M interrupt. En = 1 : Accesses via Region M address space to Bit N in any DMA Channel Register are allowed. Reads will return the value from Bit N and writes will modify the state of bit N. Enabled interrupt bits for bit N do contribute to the generation of the TPCC region M interrupt.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E31	E30	E29	E28	E27	E26	E25	E24	E23	E22	E21	E20	E19	E18	E17	E16	E15	E14	E13	E12	E11	E10	E9	E8	E7	E6	E5	E4	E3	E2	E1	E0

Bits	Field Name	Description	Type	Reset
31	E31	DMA Region Access enable for Region M, bit #31	RW	0x0
30	E30	DMA Region Access enable for Region M, bit #30	RW	0x0
29	E29	DMA Region Access enable for Region M, bit #29	RW	0x0
28	E28	DMA Region Access enable for Region M, bit #28	RW	0x0
27	E27	DMA Region Access enable for Region M, bit #27	RW	0x0
26	E26	DMA Region Access enable for Region M, bit #26	RW	0x0
25	E25	DMA Region Access enable for Region M, bit #25	RW	0x0
24	E24	DMA Region Access enable for Region M, bit #24	RW	0x0
23	E23	DMA Region Access enable for Region M, bit #23	RW	0x0
22	E22	DMA Region Access enable for Region M, bit #22	RW	0x0
21	E21	DMA Region Access enable for Region M, bit #21	RW	0x0
20	E20	DMA Region Access enable for Region M, bit #20	RW	0x0
19	E19	DMA Region Access enable for Region M, bit #19	RW	0x0
18	E18	DMA Region Access enable for Region M, bit #18	RW	0x0
17	E17	DMA Region Access enable for Region M, bit #17	RW	0x0
16	E16	DMA Region Access enable for Region M, bit #16	RW	0x0
15	E15	DMA Region Access enable for Region M, bit #15	RW	0x0
14	E14	DMA Region Access enable for Region M, bit #14	RW	0x0
13	E13	DMA Region Access enable for Region M, bit #13	RW	0x0
12	E12	DMA Region Access enable for Region M, bit #12	RW	0x0
11	E11	DMA Region Access enable for Region M, bit #11	RW	0x0
10	E10	DMA Region Access enable for Region M, bit #10	RW	0x0
9	E9	DMA Region Access enable for Region M, bit #9	RW	0x0
8	E8	DMA Region Access enable for Region M, bit #8	RW	0x0
7	E7	DMA Region Access enable for Region M, bit #7	RW	0x0
6	E6	DMA Region Access enable for Region M, bit #6	RW	0x0
5	E5	DMA Region Access enable for Region M, bit #5	RW	0x0
4	E4	DMA Region Access enable for Region M, bit #4	RW	0x0
3	E3	DMA Region Access enable for Region M, bit #3	RW	0x0
2	E2	DMA Region Access enable for Region M, bit #2	RW	0x0
1	E1	DMA Region Access enable for Region M, bit #1	RW	0x0
0	E0	DMA Region Access enable for Region M, bit #0	RW	0x0

Table 16-162. Register Call Summary for Register EDMA_TPCC_DRAEM_k

Enhanced DMA

- Region Overview: [0] [1]
- Channel Controller Regions: [2] [3] [4] [5] [6] [7]
- Transfer Completion Interrupts: [8] [9] [10] [11] [12] [13] [14] [15] [16] [17] [18] [19] [20]
- Interrupt Evaluation Operations: [21]
- Active Memory Protection: [22] [23] [24]
- Setting Up an EDMA Transfer: [25]
- EDMA Debug Checklist: [26] [27]
- EDMA Programming Tips: [28] [29] [30] [31]
- EDMA Register Summary: [32] [33] [34] [35] [36] [37]

Table 16-163. EDMA_TPCC_DRAEHM_k

Address Offset	0x0000 0344 + (0x8 * k)		
Physical Address	0x4330 0344 + (0x8 * k) 0x40D1 0344 + (0x8 * k) 0x4151 0344 + (0x8 * k) 0x01D1 0344 + (0x8 * k)	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC
Description	DMA Region Access enable for bit N in Region M: En = 0 : Accesses via Region M address space to Bit N in any DMA Channel Register are not allowed. Reads will return 'b0 on Bit N and writes will not modify the state of bit N. Enabled interrupt bits for bit N do not contribute to the generation of the TPCC region M interrupt. En = 1 : Accesses via Region M address space to Bit N in any DMA Channel Register are allowed. Reads will return the value from Bit N and writes will modify the state of bit N. Enabled interrupt bits for bit N do contribute to the generation of the TPCC region M interrupt. En = 0 : Accesses via Region M address space to Bit N in any DMA Channel Register are not allowed. Reads will return 'b0 on Bit N and writes will not modify the state of bit N. Enabled interrupt bits for bit N do not contribute to the generation of the TPCC region M interrupt. En = 1 : Accesses via Region M address space to Bit N in any DMA Channel Register are allowed. Reads will return the value from Bit N and writes will modify the state of bit N. Enabled interrupt bits for bit N do contribute to the generation of the TPCC region M interrupt.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E63	E62	E61	E60	E59	E58	E57	E56	E55	E54	E53	E52	E51	E50	E49	E48	E47	E46	E45	E44	E43	E42	E41	E40	E39	E38	E37	E36	E35	E34	E33	E32

Bits	Field Name	Description	Type	Reset
31	E63	DMA Region Access enable for Region M, bit #63	RW	0x0
30	E62	DMA Region Access enable for Region M, bit #62	RW	0x0
29	E61	DMA Region Access enable for Region M, bit #61	RW	0x0
28	E60	DMA Region Access enable for Region M, bit #60	RW	0x0
27	E59	DMA Region Access enable for Region M, bit #59	RW	0x0
26	E58	DMA Region Access enable for Region M, bit #58	RW	0x0
25	E57	DMA Region Access enable for Region M, bit #57	RW	0x0
24	E56	DMA Region Access enable for Region M, bit #56	RW	0x0
23	E55	DMA Region Access enable for Region M, bit #55	RW	0x0
22	E54	DMA Region Access enable for Region M, bit #54	RW	0x0
21	E53	DMA Region Access enable for Region M, bit #53	RW	0x0
20	E52	DMA Region Access enable for Region M, bit #52	RW	0x0
19	E51	DMA Region Access enable for Region M, bit #51	RW	0x0
18	E50	DMA Region Access enable for Region M, bit #50	RW	0x0
17	E49	DMA Region Access enable for Region M, bit #49	RW	0x0
16	E48	DMA Region Access enable for Region M, bit #48	RW	0x0
15	E47	DMA Region Access enable for Region M, bit #47	RW	0x0
14	E46	DMA Region Access enable for Region M, bit #46	RW	0x0
13	E45	DMA Region Access enable for Region M, bit #45	RW	0x0
12	E44	DMA Region Access enable for Region M, bit #44	RW	0x0
11	E43	DMA Region Access enable for Region M, bit #43	RW	0x0
10	E42	DMA Region Access enable for Region M, bit #42	RW	0x0
9	E41	DMA Region Access enable for Region M, bit #41	RW	0x0
8	E40	DMA Region Access enable for Region M, bit #40	RW	0x0
7	E39	DMA Region Access enable for Region M, bit #39	RW	0x0
6	E38	DMA Region Access enable for Region M, bit #38	RW	0x0
5	E37	DMA Region Access enable for Region M, bit #37	RW	0x0
4	E36	DMA Region Access enable for Region M, bit #36	RW	0x0
3	E35	DMA Region Access enable for Region M, bit #35	RW	0x0

Bits	Field Name	Description	Type	Reset
2	E34	DMA Region Access enable for Region M, bit #34	RW	0x0
1	E33	DMA Region Access enable for Region M, bit #33	RW	0x0
0	E32	DMA Region Access enable for Region M, bit #32	RW	0x0

Table 16-164. Register Call Summary for Register EDMA_TPCC_DRAEHM_k

Enhanced DMA
- Region Overview: [0]
- Channel Controller Regions: [1] [2] [3]
- Transfer Completion Interrupts: [4] [5] [6] [7] [8] [9] [10] [11] [12]
- Interrupt Evaluation Operations: [13]
- Setting Up an EDMA Transfer: [14]
- EDMA Debug Checklist: [15] [16] [17] [18]
- EDMA Programming Tips: [19] [20]
- EDMA Register Summary: [21] [22] [23] [24] [25] [26]

Table 16-165. EDMA_TPCC_QRAEN_k

Address Offset	0x0000 0380 + (0x4 * k)		
Physical Address	0x4330 0380 + (0x4 * k) 0x40D1 0380 + (0x4 * k) 0x4151 0380 + (0x4 * k) 0x01D1 0380 + (0x4 * k) 0x420A 0380 + (0x4 * k) 0x421A 0380 + (0x4 * k) 0x422A 0380 + (0x4 * k) 0x423A 0380 + (0x4 * k)	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	QDMA Region Access enable for bit N in Region M: En = 0 : Accesses via Region M address space to Bit N in any QDMA Channel Register are not allowed. Reads will return 'b0 on Bit N and writes will not modify the state of bit N. Enabled interrupt bits for bit N do not contribute to the generation of the TPCC region M interrupt. En = 1 : Accesses via Region M address space to Bit N in any QDMA Channel Register are allowed. Reads will return the value from Bit N and writes will modify the state of bit N. Enabled interrupt bits for bit N do contribute to the generation of the TPCC region n interrupt.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	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

Bits	Field Name	Description	Type	Reset
31:8	RESERVED	Reserved	R	0x0
7	E7	QDMA Region Access enable for Region M, bit #7	RW	0x0
6	E6	QDMA Region Access enable for Region M, bit #6	RW	0x0
5	E5	QDMA Region Access enable for Region M, bit #5	RW	0x0
4	E4	QDMA Region Access enable for Region M, bit #4	RW	0x0
3	E3	QDMA Region Access enable for Region M, bit #3	RW	0x0
2	E2	QDMA Region Access enable for Region M, bit #2	RW	0x0
1	E1	QDMA Region Access enable for Region M, bit #1	RW	0x0
0	E0	QDMA Region Access enable for Region M, bit #0	RW	0x0

Table 16-166. Register Call Summary for Register EDMA_TPCC_QRAEN_k

Enhanced DMA
- Region Overview: [0] [1]
- Channel Controller Regions: [2] [3] [4]
- EDMA Programming Tips: [5]
- EDMA Register Summary: [6] [7] [8] [9] [10] [11]

Table 16-167. EDMA_TPCC_Q0E_p

Address Offset	0x0000 0400 + (0x4 * I)		
Physical Address	0x4330 0400 + (0x4 * p) 0x40D1 0400 + (0x4 * p) 0x4151 0400 + (0x4 * p) 0x01D1 0400 + (0x4 * p)	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC
Description	Event Queue Entries Diagram for Queue 0 - Entry 0 through Entry 15		
Type	R		

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

Bits	Field Name	Description	Type	Reset
31:8	RESERVED	Reserved	R	0x0
7:6	ETYPE	Event Type: Specifies the specific Event Type for the given entry in the Event Queue.	R	0x0
5:0	ENUM	Event Number: Specifies the specific Event Number for the given entry in the Event Queue. For DMA Channel events (EDMA_TPCC_ER/EDMA_TPCC_ESR/EDMA_TPCC_CER), ENUM will range between 0 and NUM_DMACH (up to 63). For QDMA Channel events (EDMA_TPCC_QER), ENUM will range between 0 and NUM_QDMACH (up to 7).	R	0x0

Table 16-168. Register Call Summary for Register EDMA_TPCC_Q0E_p

Enhanced DMA
- Event Queue(s): [0]
- EDMA Register Summary: [1] [2] [3] [4] [5] [6]

Table 16-169. EDMA_TPCC_Q1E_p

Address Offset	0x0000 0440 + (0x4 * I)		
Physical Address	0x4330 0440 + (0x4 * p) 0x40D1 0440 + (0x4 * p) 0x4151 0440 + (0x4 * p) 0x01D1 0440 + (0x4 * p)	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC
Description	Event Queue Entries Diagram for Queue 1 - Entry 0 through Entry 15		
Type	R		

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

Bits	Field Name	Description	Type	Reset
31:8	RESERVED	Reserved	R	0x0
7:6	ETYPE	Event Type: Specifies the specific Event Type for the given entry in the Event Queue.	R	0x0
5:0	ENUM	Event Number: Specifies the specific Event Number for the given entry in the Event Queue. For DMA Channel events (EDMA_TPCC_ER / EDMA_TPCC_ESR / EDMA_TPCC_CER), ENUM will range between 0 and NUM_DMACH (up to 63). For QDMA Channel events (EDMA_TPCC_QER), ENUM will range between 0 and NUM_QDMACH (up to 7).	R	0x0

Table 16-170. Register Call Summary for Register EDMA_TPCC_Q1E_p

Enhanced DMA
- Event Queue(s): [0]
- EDMA Register Summary: [1] [2] [3] [4] [5] [6]

Table 16-171. EDMA_TPCC_QSTATN_i

Address Offset	0x0000 0600 + (0x4 * i)		
Physical Address	0x4330 0600 + (0x4 * i) 0x40D1 0600 + (0x4 * i) 0x4151 0600 + (0x4 * i) 0x01D1 0600 + (0x4 * i) 0x420A 0600 + (0x4 * i) 0x421A 0600 + (0x4 * i) 0x422A 0600 + (0x4 * i) 0x423A 0600 + (0x4 * i)	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	QSTATn Register Set		
Type	R		

31:25	24	23:21	20:16	15:13	12:8	7:4	3:0
RESERVED	THRXCD	RESERVED	WM	RESERVED	NUMVAL	RESERVED	STRTPTR

Bits	Field Name	Description	Type	Reset
31:25	RESERVED	Reserved	R Returns 0's	0x0
24	THRXCD	Threshold Exceeded 0x0 : Threshold specified by QWMTHR(A|B).Qn has not been exceeded. 0x1 : Threshold specified by QWMTHR(A|B).Qn has been exceeded. THRXCD is cleared via EDMA_TPCC_CCERR. WMCLRn bit.	R	0x0
23:21	RESERVED	Reserved	R Returns 0's	0x0

Bits	Field Name	Description	Type	Reset
20:16	WM	Watermark for Maximum Queue Usage: Watermark tracks the most entries that have been in QueueN since reset or since the last time that the watermark (WM) was cleared. QSTATn. WM is cleared via EDMA_TPCC_CCERR.WMCLRn bit. Legal values: 0x0: empty 0x10: full	R	0x0
15:13	RESERVED	Reserved	Returns 0's	0x0
12:8	NUMVAL	Number of Valid Entries in QueueN: Represents the total number of entries residing in the Queue Manager FIFO at a given instant. Always enabled. Legal values: = 0x0 (empty) to 0x10 (full) 0x0: empty 0x10: full	R	0x0
7:4	RESERVED	Reserved	Returns 0's	0x0
3:0	STRTPTR	Start Pointer: Represents the offset to the head entry of QueueN, in units of *entries*. Always enabled. Legal values: 0x0: 0th entry 0xF: 15th entry	R	0x0

Table 16-172. Register Call Summary for Register EDMA_TPCC_QSTATN_i

Enhanced DMA

- Queue RAM Debug Visibility: [0] [1] [2] [3] [4] [5] [6]
- Queue Resource Tracking: [7] [8] [9]
- EDMA Register Summary: [10] [11] [12] [13] [14] [15]

Table 16-173. EDMA_TPCC_QWMTHRA

Address Offset	0x0000 0620		
Physical Address	0x4330 0620 0x40D1 0620 0x4151 0620 0x01D1 0620 0x420A 0620 0x421A 0620 0x422A 0620 0x423A 0620	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Queue Threshold A, for Q[3:0]: EDMA_TPCC_CCERR.QTHRXCDn and QSTATn[24] THRXCD error bit is set when the number of Events in QueueN at an instant in time (visible via QSTATn[12:8] NUMVAL) equals or exceeds the value specified by EDMA_TPCC_QWMTHRA.Qn. Legal values = 0x0 (ever used?) to 0x10 (ever full?) A value of 0x11 disables threshold errors.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED			Q3					RESERVED			Q2					RESERVED			Q1					RESERVED			Q0				

Bits	Field Name	Description	Type	Reset
31:29	RESERVED	Reserved	R	0x0
28:24	Q3	Queue Threshold for Q3 value	RW	0x10
23:21	RESERVED	Reserved	R	0x0
20:16	Q2	Queue Threshold for Q2 value	RW	0x10
15:13	RESERVED	Reserved	R	0x0
12:8	Q1	Queue Threshold for Q1 value	RW	0x10
7:5	RESERVED	Reserved	R	0x0
4:0	Q0	Queue Threshold for Q0 value	RW	0x10

Table 16-174. Register Call Summary for Register EDMA_TPCC_QWMTHRA

Enhanced DMA
- Queue Resource Tracking: [0]
- EDMA Register Summary: [1] [2] [3] [4] [5] [6]
- EDMA Register Description: [7]

Table 16-175. EDMA_TPCC_QWMTHRB

Address Offset	0x0000 0624		
Physical Address	0x4330 0624 0x40D1 0624 0x4151 0624 0x01D1 0624 0x420A 0624 0x421A 0624 0x422A 0624 0x423A 0624	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Queue Threshold B, for Q[7:4]: EDMA_TPCC_CCERR.QTHRXCDn and QSTATn[24]THRXCD error bit is set when the number of Events in QueueN at an instant in time (visible via QSTATn[12:8] NUMVAL) equals or exceeds the value specified by QWMTHRB.Qn. Legal values = 0x0 (ever used?) to 0x10 (ever full?) A value of 0x11 disables threshold errors.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED			Q7					RESERVED			Q6					RESERVED			Q5					RESERVED			Q4				

Bits	Field Name	Description	Type	Reset
31:29	RESERVED	Reserved	R	0x0
28:24	Q7	Queue Threshold for Q7 value (unused in the context of IVAHD)	RW	0x10
23:21	RESERVED	Reserved	R	0x0
20:16	Q6	Queue Threshold for Q6 value (unused in the context of IVAHD)	RW	0x10
15:13	RESERVED	Reserved	R	0x0
12:8	Q5	Queue Threshold for Q5 value (unused in the context of IVAHD)	RW	0x10
7:5	RESERVED	Reserved	R	0x0
4:0	Q4	Queue Threshold for Q4 value (unused in the context of IVAHD)	RW	0x10

Table 16-176. Register Call Summary for Register EDMA_TPCC_QWMTHRB

Enhanced DMA

- EDMA Register Summary: [0] [1] [2] [3] [4] [5]

Table 16-177. EDMA_TPCC_CCSTAT

Address Offset	0x0000 0640		
Physical Address	0x4330 0640 0x40D1 0640 0x4151 0640 0x01D1 0640 0x420A 0640 0x421A 0640 0x422A 0640 0x423A 0640	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	CC Status Register		
Type	R		

31:24	23	22	21	20	19	18	17	16	15	14:8	7:5	4	3	2	1	0
RESERVED	QUEACTV7	QUEACTV6	QUEACTV5	QUEACTV4	QUEACTV3	QUEACTV2	QUEACTV1	QUEACTV0	RESERVED	COMPACTV	RESERVED	ACTV	RESERVED	TRACTV	QEVTACTV	EVTACTV

Bits	Field Name	Description	Type	Reset
31:24	RESERVED	reads return 0's	R	0x0
23	QUEACTV7	Queue 7 Active 0x0: No Evts are queued in Q7 0x1: At least one TR is queued in Q7.	R	0x0
22	QUEACTV6	Queue 6 Active 0x0: No Evts are queued in Q6. 0x1: At least one TR is queued in Q6.	R	0x0
21	QUEACTV5	Queue 5 Active 0x0: No Evts are queued in Q5 0x1: At least one TR is queued in Q5.	R	0x0
20	QUEACTV4	Queue 4 Active 0x0: No Evts are queued in Q4. 0x1: At least one TR is queued in Q4.	R	0x0
19	QUEACTV3	Queue 3 Active 0x0: No Evts are queued in Q3. 0x1: At least one TR is queued in Q3.	R	0x0
18	QUEACTV2	Queue 2 Active QUEACTV2 = 0 : No Evts are queued in Q2. QUEACTV2 = 1 : At least one TR is queued in Q2. 0x0: 0x1:	R	0x0
17	QUEACTV1	Queue 1 Active 0x0: No Evts are queued in Q1. 0x1: At least one TR is queued in Q1.	R	0x0
16	QUEACTV0	Queue 0 Active 0x0: No Evts are queued in Q0. 0x1: At least one TR is queued in Q0.	R	0x0

Bits	Field Name	Description	Type	Reset
15:14	RESERVED	Reserved	R reads return 0's	0x0
13:8	COMPACTV	Completion Request Active: Counter that tracks the total number of completion requests submitted to the TC. The counter increments when a TR is submitted with TCINTEN or TCCHEN set to '1'. The counter decrements for every valid completion code received from any of the external TCs. The CC will not service new TRs if COMPACTV count is already at the limit. 0x0: No completion requests outstanding. 0x1: Total of '1' completion request outstanding. ... 0x3F: Total of 63 completion requests are outstanding. No additional TRs will be submitted until count is less than 63.	R	0x0
7:5	RESERVED	reads return 0's	R	0x0
4	ACTV	Channel Controller Active Channel Controller Active is a logical-OR of each of the *ACTV signals. The ACTV bit must remain high through the life of a: 0x0: Channel is idle. 0x1: Channel is busy.	R	0x0
3	RESERVED	reads return 0's	R	0x0
2	TRACTV	Transfer Request Active TRACTV = 0 : Transfer Request processing/submission logic is inactive. TRACTV = 1 : Transfer Request processing/submission logic is active. 0x0: 0x1:	R	0x0
1	QEVTACTV	QDMA Event Active 0x0: No enabled QDMA Events are active within the CC. 0x1: At least one enabled DMA Event (EDMA_TPCC_ER, EDMA_TPCC_EER, EDMA_TPCC_ESR, EDMA_TPCC_CER) is active within the CC.	R	0x0
0	EVTACTV	DMA Event Active 0x0: No enabled DMA Events are active within the CC. 0x1: At least one enabled DMA Event (EDMA_TPCC_ER, EDMA_TPCC_EER, EDMA_TPCC_ESR, EDMA_TPCC_CER) is active within the CC.	R	0x0

Table 16-178. Register Call Summary for Register EDMA_TPCC_CCSTAT

Enhanced DMA

- EDMA Register Summary: [0] [1] [2] [3] [4] [5]

Table 16-179. EDMA_TPCC_AETCTL

Address Offset	0x0000 0700		
Physical Address	0x4330 0700 0x40D1 0700 0x4151 0700 0x01D1 0700 0x420A 0700 0x421A 0700 0x422A 0700 0x423A 0700	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC

Table 16-179. EDMA_TPCC_AETCTL (continued)

Description	Advanced Event Trigger Control
Type	RW

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN	RESERVED																	ENDINT						RESERVED	TYPE	STRTEVT					

Bits	Field Name	Description	Type	Reset
31	EN	AET Enable 0x0: AET event generation is disabled. 0x1: AET event generation is enabled.	RW	0x0
30:14	RESERVED	Reserved	R	0x0
13:8	ENDINT	AET End Interrupt: Dictates the completion interrupt number that will force the tpcc_aet signal to be deasserted (low)	RW	0x0
7	RESERVED	Reserved	R	0x0
6	TYPE	AET Event Type 0x0: Event specified by STARTEVT applies to DMA Events (set by EDMA_TPCC_ER, EDMA_TPCC_ESR, or EDMA_TPCC_CER) 0x1: Event specified by STARTEVT applies to QDMA Events	RW	0x0
5:0	STRTEVT	AET Start Event: Dictates the Event Number that will force the tpcc_aet signal to be asserted (high)	RW	0x0

Table 16-180. Register Call Summary for Register EDMA_TPCC_AETCTL

Enhanced DMA

- EDMA Register Summary: [0] [1] [2] [3] [4] [5]

Table 16-181. EDMA_TPCC_AETSTAT

Address Offset	0x0000 0704		
Physical Address	0x4330 0704 0x40D1 0704 0x4151 0704 0x01D1 0704 0x420A 0704 0x421A 0704 0x422A 0704 0x423A 0704	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Advanced Event Trigger Stat		
Type	R		

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

Bits	Field Name	Description	Type	Reset
31:1	RESERVED	Reserved	R Return 0's	0x0
0	STAT	AET Status 0x0: tpcc_aet is currently low. 0x1: tpcc_aet is currently high.	R	0x0

Table 16-182. Register Call Summary for Register EDMA_TPCC_AETSTAT

Enhanced DMA
- EDMA Register Summary: [0] [1] [2] [3] [4] [5]
- EDMA Register Description: [6]

Table 16-183. EDMA_TPCC_AETCMD

Address Offset	0x0000 0708		
Physical Address	0x4330 0708 0x40D1 0708 0x4151 0708 0x01D1 0708 0x420A 0708 0x421A 0708 0x422A 0708 0x423A 0708	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	AET Command		
Type	W		

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

Bits	Field Name	Description	Type	Reset
31:1	RESERVED	Reserved	R	0x0
0	CLR	AET Clear command CPU writes 0x0 has no effect. CPU writes 0x1 to the CLR bit causes the tpcc_aet output signal and EDMA_TPCC_AETSTAT[0]STAT register to be cleared.	W	0x0

Table 16-184. Register Call Summary for Register EDMA_TPCC_AETCMD

Enhanced DMA
- EDMA Register Summary: [0] [1] [2] [3] [4] [5]

Table 16-185. EDMA_TPCC_MPFAR

Address Offset	0x0000 0800		
Physical Address	0x4330 0800 0x40D1 0800 0x4151 0800 0x01D1 0800 0x420A 0800 0x421A 0800 0x422A 0800 0x423A 0800	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	MMemory Protection Fault Address		
Type	R		

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

Bits	Field Name	Description	Type	Reset
31:0	FADDR	Fault Address: 32-bit read-only status register containing the faulting address when a mMemory protection violation is detected. This register can only be cleared via the EDMA_TPCC_MPFCR.	R	0x0

Table 16-186. Register Call Summary for Register EDMA_TPCC_MPFAR

Enhanced DMA

- EDMA Register Summary: [0] [1] [2] [3] [4] [5]
- EDMA Register Description: [6]

Table 16-187. EDMA_TPCC_MPFSR

Address Offset	0x0000 0804		
Physical Address	0x4330 0804 0x40D1 0804 0x4151 0804 0x01D1 0804 0x420A 0804 0x421A 0804 0x422A 0804 0x423A 0804	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Memory Protection Fault Status Register		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																			FID				RESERVED			SRE	SWE	SXE	URE	UWE	UXE

Bits	Field Name	Description	Type	Reset
31:13	RESERVED	Reserved	R Returns 0	0x0
12:9	FID	Faulted ID: FID register contains valid info if any of the MP error bits (UXE, UWE, URE, SXE, SWE, SRE) are non-zero (i.e., if an error has been detected.) The FID field contains the VBus PrivID for the specific request/requestor that resulted in a MP Error.	R	0x0
8:6	RESERVED	Reserved	R Returns 0	0x0
5	SRE	Supervisor Read Error 0x0: No error detected. 0x1: Supervisor level task attempted to Read from a MP Page without SR permissions.	R	0x0
4	SWE	Supervisor Write Error 0x0: No error detected. 0x1: Supervisor level task attempted to Write to a MP Page without SW permissions.	R	0x0

Bits	Field Name	Description	Type	Reset
3	SXE	Supervisor Execute Error 0x0: No error detected. 0x1: Supervisor level task attempted to Execute from a MP Page without SX permissions.	R	0x0
2	URE	User Read Error 0x0: No error detected. 0x1: User level task attempted to Read from a MP Page without UR permissions.	R	0x0
1	UWE	User Write Error 0x0: No error detected. 0x1: User level task attempted to Write to a MP Page without UW permissions.	R	0x0
0	UXE	User Execute Error 0x0: No error detected 0x1: User level task attempted to Execute from a MP Page without UX permissions.	R	0x0

Table 16-188. Register Call Summary for Register EDMA_TPCC_MPFSR

Enhanced DMA
- EDMA Register Summary: [0] [1] [2] [3] [4] [5]
- EDMA Register Description: [6]

Table 16-189. EDMA_TPCC_MPFCR

Address Offset	0x0000 0808		
Physical Address	0x4330 0808 0x40D1 0808 0x4151 0808 0x01D1 0808 0x420A 0808 0x421A 0808 0x422A 0808 0x423A 0808	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Memory Protection Fault Command Register		
Type	W		

31–1	0
RESERVED	MPFCLR

Bits	Field Name	Description	Type	Reset
31:1	RESERVED	Reserved	R	0x0
0	MPFCLR	Fault Clear register CPU writes 0x0: has no effect CPU writes 0x1: to the MPFCLR bit causes any error conditions stored in EDMA_TPCC_MPFAR and EDMA_TPCC_MPFSR registers to be cleared.	W	0x0

Table 16-190. Register Call Summary for Register EDMA_TPCC_MPFCR

Enhanced DMA
- EDMA Register Summary: [0] [1] [2] [3] [4] [5]
- EDMA Register Description: [6]

Table 16-191. EDMA_TPCC_MPPAG

Address Offset	0x0000 080C		
Physical Address	0x4330 080C 0x40D1 080C 0x4151 080C 0x01D1 080C 0x420A 080C 0x421A 080C 0x422A 080C 0x423A 080C	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Memory Protection Page Attribute for Global registers		
Type	RW		

31:16	15	14	13	12	11	10	9	8:6	5	4	3	2	1	0
RESERVED	AID5	AID4	AID3	AID2	AID1	AID0	EXT	RESERVED	SR	SW	SX	UR	UW	UX

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Reserved	R	0x0
15	AID5	Allowed ID 5 0x0: VBus requests with PrivID == '5' are not allowed regardless of permission settings (UW, UR, SW, SR).0 0x1: VBus requests with PrivID == '5' are permitted if access type is allowed as defined by permission settings (UW, UR, SW, SR).	RW	0x1
14	AID4	Allowed ID 4 0x0: VBus requests with PrivID == '4' are not allowed regardless of permission settings (UW, UR, SW, SR). 0x1: VBus requests with PrivID == '4' are permitted if access type is allowed as defined by permission settings (UW, UR, SW, SR).	RW	0x1
13	AID3	Allowed ID 3 0x0: VBus requests with PrivID == '3' are not allowed regardless of permission settings (UW, UR, SW, SR). 0x1: VBus requests with PrivID == '3' are permitted if access type is allowed as defined by permission settings (UW, UR, SW, SR).	RW	0x1
12	AID2	Allowed ID 2 0x0: VBus requests with PrivID == '2' are not allowed regardless of permission settings (UW, UR, SW, SR). 0x1: VBus requests with PrivID == '2' are permitted if access type is allowed as defined by permission settings (UW, UR, SW, SR).	RW	0x1
11	AID1	Allowed ID 1 0x0: VBus requests with PrivID == '1' are not allowed regardless of permission settings (UW, UR, SW, SR). 0x1: VBus requests with PrivID == '1' are permitted if access type is allowed as defined by permission settings (UW, UR, SW, SR).	RW	0x1
10	AID0	Allowed ID 0 0x0: VBus requests with PrivID == '0' are not allowed regardless of permission settings (UW, UR, SW, SR). 0x1: VBus requests with PrivID == '0' are permitted if access type is allowed as defined by permission settings (UW, UR, SW, SR).	RW	0x1

Bits	Field Name	Description	Type	Reset
9	EXT	External Allowed ID 0x0: VBus requests with PrivID = '6' are not allowed regardless of permission settings (UW, UR, SW, SR). 0x1: VBus requests with PrivID = '6' are permitted if access type is allowed as defined by permission settings (UW, UR, SW, SR).	RW	0x1
8:6	RESERVED	Reserved	R	0x1
5	SR	Supervisor Read permission 0x0: Supervisor read accesses are not allowed 0x1: Supervisor write accesses are allowed	RW	0x1
4	SW	Supervisor Write permission 0x0: Supervisor write accesses are not allowed 0x1: Supervisor write accesses are allowed	RW	0x1
3	SX	Supervisor Execute permission 0x0: Supervisor execute accesses are not allowed 0x1: Supervisor execute accesses are allowed	RW	0x0
2	UR	User Read permission 0x0: User read accesses are not allowed 0x1: User write accesses are allowed	RW	0x1
1	UW	User Write permission 0x0: User write accesses are not allowed 0x1: User write accesses are allowed	RW	0x1
0	UX	User Execute permission 0x0: User execute accesses are not allowed 0x1: User execute accesses are allowed	RW	0x0

Table 16-192. Register Call Summary for Register EDMA_TPCC_MPPAG

Enhanced DMA

- Channel Controller Regions: [0]
- Active Memory Protection: [1] [2] [3] [4]
- EDMA Register Summary: [5] [6] [7] [8] [9] [10]

Table 16-193. EDMA_TPCC_MPPAN_k

Address Offset	0x0000 0810 + (0x4 * k)		
Physical Address	0x4330 0810 + (0x4 * k) 0x40D1 0810 + (0x4 * k) 0x4151 0810 + (0x4 * k) 0x01D1 0810 + (0x4 * k) 0x420A 0810 + (0x4 * k) 0x421A 0810 + (0x4 * k) 0x422A 0810 + (0x4 * k) 0x423A 0810 + (0x4 * k)	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	P Permission Attribute for DMA Region n		
Type	RW		

31:16	15	14	13	12	11	10	9	8:6	5	4	3	2	1	0
RESERVED	AID5	AID4	AID3	AID2	AID1	AID0	EXT	RESERVED	SR	SW	SX	UR	UW	UX

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Reserved	R	0x0
15	AID5	Allowed ID 5 0x0: VBus requests with PrivID == '5' are not allowed regardless of permission settings (UW, UR, SW, SR). 0x1: VBus requests with PrivID == '5' are permitted if access type is allowed as defined by permission settings (UW, UR, SW, SR).	RW	0x1
14	AID4	Allowed ID 4 0x0: VBus requests with PrivID == '4' are not allowed regardless of permission settings (UW, UR, SW, SR). 0x1: VBus requests with PrivID == '4' are permitted if access type is allowed as defined by permission settings (UW, UR, SW, SR).	RW	0x1
13	AID3	Allowed ID 3 0x0: VBus requests with PrivID == '3' are not allowed regardless of permission settings (UW, UR, SW, SR). 0x1: VBus requests with PrivID == '3' are permitted if access type is allowed as defined by permission settings (UW, UR, SW, SR).	RW	0x1
12	AID2	Allowed ID 2 0x0: VBus requests with PrivID == '2' are not allowed regardless of permission settings (UW, UR, SW, SR). 0x1: VBus requests with PrivID == '2' are permitted if access type is allowed as defined by permission settings (UW, UR, SW, SR).	RW	0x1
11	AID1	Allowed ID 1 0x0: VBus requests with PrivID == '1' are not allowed regardless of permission settings (UW, UR, SW, SR). 0x1: VBus requests with PrivID == '1' are permitted if access type is allowed as defined by permission settings (UW, UR, SW, SR).	RW	0x1
10	AID0	Allowed ID 0: AID0 = 0 : VBus requests with PrivID == '0' are not allowed regardless of permission settings (UW, UR, SW, SR). AID0 = 1 : VBus requests with PrivID == '0' are permitted if access type is allowed as defined by permission settings (UW, UR, SW, SR). 0x0: 0x1:	RW	0x1
9	EXT	External Allowed ID 0x0: VBus requests with PrivID = '6' are not allowed regardless of permission settings (UW, UR, SW, SR). 0x1: VBus requests with PrivID = '6' are permitted if access type is allowed as defined by permission settings (UW, UR, SW, SR).	RW	0x1
8:6	RESERVED	Reserved	R	0x0
5	SR	Supervisor Read permission 0x0: Supervisor read accesses are not allowed 0x1: Supervisor write accesses are allowed	RW	0x1
4	SW	Supervisor Write permission 0x0: Supervisor write accesses are not allowed 0x1: Supervisor write accesses are allowed	RW	0x1
3	SX	Supervisor Execute permission 0x0: Supervisor execute accesses are not allowed 0x1: Supervisor execute accesses are allowed	RW	0x0

Bits	Field Name	Description	Type	Reset
2	UR	User Read permission 0x0: User read accesses are not allowed 0x1: User write accesses are allowed	RW	0x1
1	UW	User Write permission 0x0: User write accesses are not allowed 0x1: User write accesses are allowed	RW	0x1
0	UX	User Execute permission 0x0: User execute accesses are not allowed 0x0: User execute accesses are allowed	RW	0x0

Table 16-194. Register Call Summary for Register EDMA_TPCC_MPPAN_k

Enhanced DMA

- Channel Controller Regions: [0]
- Active Memory Protection: [1] [2] [3] [4] [5] [6] [7] [8] [9] [10] [11] [12] [13] [14] [15] [16] [17] [18] [19] [20] [21] [22] [23] [24] [25] [26] [27] [28] [29] [30] [31] [32] [33] [34] [35]
- Proxy Memory Protection: [36] [37] [38] [39]
- EDMA Register Summary: [40] [41] [42] [43] [44] [45]

Table 16-195. EDMA_TPCC_ER

Address Offset	0x0000 1000		
Physical Address	0x4330 1000 0x40D1 1000 0x4151 1000 0x01D1 1000 0x420A 1000 0x421A 1000 0x422A 1000 0x423A 1000	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Event Register: If EDMA_TPCC_ER.En bit is set and the EDMA_TPCC_EER.En bit is also set, then the corresponding DMA channel is prioritized vs. other pending DMA events for submission to the TC. EDMA_TPCC_ER.En bit is set when the input event #n transitions from inactive (low) to active (high), regardless of the state of EDMA_TPCC_EER.En bit. EDMA_TPCC_ER.En bit is cleared when the corresponding event is prioritized and serviced. If the EDMA_TPCC_ER.En bit is already set and a new inactive to active transition is detected on the input event #n input AND the corresponding bit in the EDMA_TPCC_EER register is set, then the corresponding bit in the Event Missed Register is set. Event N can be cleared via sw by writing a '1' to the EDMA_TPCC_ECR pseudo-register.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E31	E30	E29	E28	E27	E26	E25	E24	E23	E22	E21	E20	E19	E18	E17	E16	E15	E14	E13	E12	E11	E10	E9	E8	E7	E6	E5	E4	E3	E2	E1	E0

Bits	Field Name	Description	Type	Reset
31	E31	Event #31	R	0x0
30	E30	Event #30	R	0x0
29	E29	Event #29	R	0x0
28	E28	Event #28	R	0x0
27	E27	Event #27	R	0x0
26	E26	Event #26	R	0x0
25	E25	Event #25	R	0x0
24	E24	Event #24	R	0x0
23	E23	Event #23	R	0x0

Bits	Field Name	Description	Type	Reset
22	E22	Event #22	R	0x0
21	E21	Event #21	R	0x0
20	E20	Event #20	R	0x0
19	E19	Event #19	R	0x0
18	E18	Event #18	R	0x0
17	E17	Event #17	R	0x0
16	E16	Event #16	R	0x0
15	E15	Event #15	R	0x0
14	E14	Event #14	R	0x0
13	E13	Event #13	R	0x0
12	E12	Event #12	R	0x0
11	E11	Event #11	R	0x0
10	E10	Event #10	R	0x0
9	E9	Event #9	R	0x0
8	E8	Event #8	R	0x0
7	E7	Event #7	R	0x0
6	E6	Event #6	R	0x0
5	E5	Event #5	R	0x0
4	E4	Event #4	R	0x0
3	E3	Event #3	R	0x0
2	E2	Event #2	R	0x0
1	E1	Event #1	R	0x0
0	E0	Event #0	R	0x0

Table 16-196. Register Call Summary for Register EDMA_TPCC_ER

Enhanced DMA

- DMA Channel: [0] [1] [2] [3] [4] [5] [6] [7] [8] [9]
- Region Overview: [10]
- Event Dataflow: [11] [12]
- Channel Priority: [13]
- Trigger Source Priority: [14] [15] [16] [17]
- Setting Up an EDMA Transfer: [18]
- EDMA Debug Checklist: [19]
- EDMA Programming Tips: [20]
- EDMA Register Summary: [21] [22] [23] [24] [25] [26]
- EDMA Register Description: [27] [28] [29] [30] [31] [32] [33] [34] [35] [36] [37] [38] [39] [40] [41] [42] [43] [44] [45] [46] [47] [48] [49] [50] [51] [52] [53] [54] [55] [56] [57] [58] [59] [60] [61] [62] [63] [64] [65]

Table 16-197. EDMA_TPCC_ERH

Address Offset	0x0000 1004		
Physical Address	0x4330 1004 0x40D1 1004 0x4151 1004 0x01D1 1004 0x420A 1004 0x421A 1004 0x422A 1004 0x423A 1004	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC

Table 16-197. EDMA_TPCC_ERH (continued)

Description	Event Register (High Part): If EDMA_TPCC_ERH.En bit is set and the EDMA_TPCC_EERH.En bit is also set, then the corresponding DMA channel is prioritized vs. other pending DMA events for submission to the TC. EDMA_TPCC_ERH.En bit is set when the input event #n transitions from inactive (low) to active (high), regardless of the state of EDMA_TPCC_EERH.En bit. EDMA_TPCC_ER.En bit is cleared when the corresponding event is prioritized and serviced. If the EDMA_TPCC_ERH.En bit is already set and a new inactive to active transition is detected on the input event #n input AND the corresponding bit in the EDMA_TPCC_EERH register is set, then the corresponding bit in the Event Missed Register is set. Event N can be cleared via sw by writing a '1' to the EDMA_TPCC_ECRH pseudo-register.
Type	R

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E63	E62	E61	E60	E59	E58	E57	E56	E55	E54	E53	E52	E51	E50	E49	E48	E47	E46	E45	E44	E43	E42	E41	E40	E39	E38	E37	E36	E35	E34	E33	E32

Bits	Field Name	Description	Type	Reset
31	E63	Event #63	R	0x0
30	E62	Event #62	R	0x0
29	E61	Event #61	R	0x0
28	E60	Event #60	R	0x0
27	E59	Event #59	R	0x0
26	E58	Event #58	R	0x0
25	E57	Event #57	R	0x0
24	E56	Event #56	R	0x0
23	E55	Event #55	R	0x0
22	E54	Event #54	R	0x0
21	E53	Event #53	R	0x0
20	E52	Event #52	R	0x0
19	E51	Event #51	R	0x0
18	E50	Event #50	R	0x0
17	E49	Event #49	R	0x0
16	E48	Event #48	R	0x0
15	E47	Event #47	R	0x0
14	E46	Event #46	R	0x0
13	E45	Event #45	R	0x0
12	E44	Event #44	R	0x0
11	E43	Event #43	R	0x0
10	E42	Event #42	R	0x0
9	E41	Event #41	R	0x0
8	E40	Event #40	R	0x0
7	E39	Event #39	R	0x0
6	E38	Event #38	R	0x0
5	E37	Event #37	R	0x0
4	E36	Event #36	R	0x0
3	E35	Event #35	R	0x0
2	E34	Event #34	R	0x0
1	E33	Event #33	R	0x0
0	E32	Event #32	R	0x0

Table 16-198. Register Call Summary for Register EDMA_TPCC_ERH

Enhanced DMA
- Region Overview: [0]
- Event Dataflow: [1] [2]
- Channel Priority: [3]
- EDMA Debug Checklist: [4]
- EDMA Programming Tips: [5]
- EDMA Register Summary: [6] [7] [8] [9] [10] [11]
- EDMA Register Description: [12] [13] [14] [15] [16] [17] [18] [19] [20] [21] [22] [23] [24] [25] [26] [27] [28] [29] [30] [31]

Table 16-199. EDMA_TPCC_ECR

Address Offset	0x0000 1008		
Physical Address	0x4330 1008 0x40D1 1008 0x4151 1008 0x01D1 1008 0x420A 1008 0x421A 1008 0x422A 1008 0x423A 1008	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Event Clear Register: CPU write of '1' to the EDMA_TPCC_ECR.En bit causes the EDMA_TPCC_ER.En bit to be cleared. CPU write of '0' has no effect.		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E31	E30	E29	E28	E27	E26	E25	E24	E23	E22	E21	E20	E19	E18	E17	E16	E15	E14	E13	E12	E11	E10	E9	E8	E7	E6	E5	E4	E3	E2	E1	E0

Bits	Field Name	Description	Type	Reset
31	E31	Event #31	W	0x0
30	E30	Event #30	W	0x0
29	E29	Event #29	W	0x0
28	E28	Event #28	W	0x0
27	E27	Event #27	W	0x0
26	E26	Event #26	W	0x0
25	E25	Event #25	W	0x0
24	E24	Event #24	W	0x0
23	E23	Event #23	W	0x0
22	E22	Event #22	W	0x0
21	E21	Event #21	W	0x0
20	E20	Event #20	W	0x0
19	E19	Event #19	W	0x0
18	E18	Event #18	W	0x0
17	E17	Event #17	W	0x0
16	E16	Event #16	W	0x0
15	E15	Event #15	W	0x0
14	E14	Event #14	W	0x0
13	E13	Event #13	W	0x0
12	E12	Event #12	W	0x0
11	E11	Event #11	W	0x0
10	E10	Event #10	W	0x0

Bits	Field Name	Description	Type	Reset
9	E9	Event #9	W	0x0
8	E8	Event #8	W	0x0
7	E7	Event #7	W	0x0
6	E6	Event #6	W	0x0
5	E5	Event #5	W	0x0
4	E4	Event #4	W	0x0
3	E3	Event #3	W	0x0
2	E2	Event #2	W	0x0
1	E1	Event #1	W	0x0
0	E0	Event #0	W	0x0

Table 16-200. Register Call Summary for Register EDMA_TPCC_ECR

Enhanced DMA

- Region Overview: [0]
- EDMA Programming Tips: [1]
- EDMA Register Summary: [2] [3] [4] [5] [6] [7]
- EDMA Register Description: [8] [9] [10] [11]

Table 16-201. EDMA_TPCC_ECRH

Address Offset	0x0000 100C		
Physical Address	0x4330 100C 0x40D1 100C 0x4151 100C 0x01D1 100C 0x420A 100C 0x421A 100C 0x422A 100C 0x423A 100C	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Event Clear Register (High Part): CPU write of '1' to the EDMA_TPCC_ECRH.En bit causes the EDMA_TPCC_ERH.En bit to be cleared. CPU write of '0' has no effect.		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E63	E62	E61	E60	E59	E58	E57	E56	E55	E54	E53	E52	E51	E50	E49	E48	E47	E46	E45	E44	E43	E42	E41	E40	E39	E38	E37	E36	E35	E34	E33	E32

Bits	Field Name	Description	Type	Reset
31	E63	Event #63	W	0x0
30	E62	Event #62	W	0x0
29	E61	Event #61	W	0x0
28	E60	Event #60	W	0x0
27	E59	Event #59	W	0x0
26	E58	Event #58	W	0x0
25	E57	Event #57	W	0x0
24	E56	Event #56	W	0x0
23	E55	Event #55	W	0x0
22	E54	Event #54	W	0x0
21	E53	Event #53	W	0x0
20	E52	Event #52	W	0x0
19	E51	Event #51	W	0x0

Bits	Field Name	Description	Type	Reset
18	E50	Event #50	W	0x0
17	E49	Event #49	W	0x0
16	E48	Event #48	W	0x0
15	E47	Event #47	W	0x0
14	E46	Event #46	W	0x0
13	E45	Event #45	W	0x0
12	E44	Event #44	W	0x0
11	E43	Event #43	W	0x0
10	E42	Event #42	W	0x0
9	E41	Event #41	W	0x0
8	E40	Event #40	W	0x0
7	E39	Event #39	W	0x0
6	E38	Event #38	W	0x0
5	E37	Event #37	W	0x0
4	E36	Event #36	W	0x0
3	E35	Event #35	W	0x0
2	E34	Event #34	W	0x0
1	E33	Event #33	W	0x0
0	E32	Event #32	W	0x0

Table 16-202. Register Call Summary for Register EDMA_TPCC_ECRH

Enhanced DMA

- Region Overview: [0]
- EDMA Programming Tips: [1]
- EDMA Register Summary: [2] [3] [4] [5] [6] [7]
- EDMA Register Description: [8] [9] [10] [11]

Table 16-203. EDMA_TPCC_ESR

Address Offset	0x0000 1010		
Physical Address	0x4330 1010 0x40D1 1010 0x4151 1010 0x01D1 1010 0x420A 1010 0x421A 1010 0x422A 1010 0x423A 1010	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Event Set Register: CPU write of '1' to the EDMA_TPCC_ESR.En bit causes the EDMA_TPCC_ER.En bit to be set. CPU write of '0' has no effect.		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E31	E30	E29	E28	E27	E26	E25	E24	E23	E22	E21	E20	E19	E18	E17	E16	E15	E14	E13	E12	E11	E10	E9	E8	E7	E6	E5	E4	E3	E2	E1	E0

Bits	Field Name	Description	Type	Reset
31	E31	Event #31	W	0x0
30	E30	Event #30	W	0x0
29	E29	Event #29	W	0x0
28	E28	Event #28	W	0x0

Bits	Field Name	Description	Type	Reset
27	E27	Event #27	W	0x0
26	E26	Event #26	W	0x0
25	E25	Event #25	W	0x0
24	E24	Event #24	W	0x0
23	E23	Event #23	W	0x0
22	E22	Event #22	W	0x0
21	E21	Event #21	W	0x0
20	E20	Event #20	W	0x0
19	E19	Event #19	W	0x0
18	E18	Event #18	W	0x0
17	E17	Event #17	W	0x0
16	E16	Event #16	W	0x0
15	E15	Event #15	W	0x0
14	E14	Event #14	W	0x0
13	E13	Event #13	W	0x0
12	E12	Event #12	W	0x0
11	E11	Event #11	W	0x0
10	E10	Event #10	W	0x0
9	E9	Event #9	W	0x0
8	E8	Event #8	W	0x0
7	E7	Event #7	W	0x0
6	E6	Event #6	W	0x0
5	E5	Event #5	W	0x0
4	E4	Event #4	W	0x0
3	E3	Event #3	W	0x0
2	E2	Event #2	W	0x0
1	E1	Event #1	W	0x0
0	E0	Event #0	W	0x0

Table 16-204. Register Call Summary for Register EDMA_TPCC_ESR

Enhanced DMA

- EDMA Features: [0]
- Third-Party Channel Controller: [1]
- Initiating a DMA Transfer: [2]
- DMA Channel: [3] [4] [5] [6] [7]
- Comparison Between DMA and QDMA Channels: [8]
- Region Overview: [9]
- Event Dataflow: [10] [11]
- Trigger Source Priority: [12] [13]
- Transfer Chaining Examples: [14]
- Setting Up an EDMA Transfer: [15]
- EDMA Register Summary: [16] [17] [18] [19] [20] [21]
- EDMA Register Description: [22] [23] [24] [25] [26] [27] [28] [29] [30]

Table 16-205. EDMA_TPCC_ESRH

Address Offset	0x0000 1014		
Physical Address	0x4330 1014 0x40D1 1014 0x4151 1014 0x01D1 1014 0x420A 1014 0x421A 1014 0x422A 1014 0x423A 1014	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Event Set Register (High Part) CPU write of '1' to the EDMA_TPCC_ESRH.En bit causes the EDMA_TPCC_ERH.En bit to be set. CPU write of '0' has no effect.		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E63	E62	E61	E60	E59	E58	E57	E56	E55	E54	E53	E52	E51	E50	E49	E48	E47	E46	E45	E44	E43	E42	E41	E40	E39	E38	E37	E36	E35	E34	E33	E32

Bits	Field Name	Description	Type	Reset
31	E63	Event #63	W	0x0
30	E62	Event #62	W	0x0
29	E61	Event #61	W	0x0
28	E60	Event #60	W	0x0
27	E59	Event #59	W	0x0
26	E58	Event #58	W	0x0
25	E57	Event #57	W	0x0
24	E56	Event #56	W	0x0
23	E55	Event #55	W	0x0
22	E54	Event #54	W	0x0
21	E53	Event #53	W	0x0
20	E52	Event #52	W	0x0
19	E51	Event #51	W	0x0
18	E50	Event #50	W	0x0
17	E49	Event #49	W	0x0
16	E48	Event #48	W	0x0
15	E47	Event #47	W	0x0
14	E46	Event #46	W	0x0
13	E45	Event #45	W	0x0
12	E44	Event #44	W	0x0
11	E43	Event #43	W	0x0
10	E42	Event #42	W	0x0
9	E41	Event #41	W	0x0
8	E40	Event #40	W	0x0
7	E39	Event #39	W	0x0
6	E38	Event #38	W	0x0
5	E37	Event #37	W	0x0
4	E36	Event #36	W	0x0
3	E35	Event #35	W	0x0
2	E34	Event #34	W	0x0
1	E33	Event #33	W	0x0
0	E32	Event #32	W	0x0

Table 16-206. Register Call Summary for Register EDMA_TPCC_ESRH

Enhanced DMA
- EDMA Features: [0]
- Third-Party Channel Controller: [1]
- Initiating a DMA Transfer: [2]
- Region Overview: [3]
- Event Dataflow: [4] [5]
- Setting Up an EDMA Transfer: [6]
- EDMA Register Summary: [7] [8] [9] [10] [11] [12]
- EDMA Register Description: [13] [14] [15] [16]

Table 16-207. EDMA_TPCC_CER

Address Offset	0x0000 1018		
Physical Address	0x4330 1018 0x40D1 1018 0x4151 1018 0x01D1 1018 0x420A 1018 0x421A 1018 0x422A 1018 0x423A 1018	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Chained Event Register: If EDMA_TPCC_CER.En bit is set (regardless of state of EDMA_TPCC_EER.En), then the corresponding DMA channel is prioritized vs. other pending DMA events for submission to the TC. EDMA_TPCC_CER.En bit is set when a chaining completion code is returned from one of the 3PTCs via the completion interface, or is generated internally via Early Completion path. EDMA_TPCC_CER.En bit is cleared when the corresponding event is prioritized and serviced. If the EDMA_TPCC_CER.En bit is already set and the corresponding chaining completion code is returned from the TC, then the corresponding bit in the Event Missed Register is set. EDMA_TPCC_CER.En cannot be set or cleared via software.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E31	E30	E29	E28	E27	E26	E25	E24	E23	E22	E21	E20	E19	E18	E17	E16	E15	E14	E13	E12	E11	E10	E9	E8	E7	E6	E5	E4	E3	E2	E1	E0

Bits	Field Name	Description	Type	Reset
31	E31	Event #31	R	0x0
30	E30	Event #30	R	0x0
29	E29	Event #29	R	0x0
28	E28	Event #28	R	0x0
27	E27	Event #27	R	0x0
26	E26	Event #26	R	0x0
25	E25	Event #25	R	0x0
24	E24	Event #24	R	0x0
23	E23	Event #23	R	0x0
22	E22	Event #22	R	0x0
21	E21	Event #21	R	0x0
20	E20	Event #20	R	0x0
19	E19	Event #19	R	0x0
18	E18	Event #18	R	0x0
17	E17	Event #17	R	0x0
16	E16	Event #16	R	0x0
15	E15	Event #15	R	0x0
14	E14	Event #14	R	0x0

Bits	Field Name	Description	Type	Reset
13	E13	Event #13	R	0x0
12	E12	Event #12	R	0x0
11	E11	Event #11	R	0x0
10	E10	Event #10	R	0x0
9	E9	Event #9	R	0x0
8	E8	Event #8	R	0x0
7	E7	Event #7	R	0x0
6	E6	Event #6	R	0x0
5	E5	Event #5	R	0x0
4	E4	Event #4	R	0x0
3	E3	Event #3	R	0x0
2	E2	Event #2	R	0x0
1	E1	Event #1	R	0x0
0	E0	Event #0	R	0x0

Table 16-208. Register Call Summary for Register EDMA_TPCC_CER

Enhanced DMA

- Third-Party Channel Controller: [0]
- Dummy Versus Null Transfer Comparison: [1]
- DMA Channel: [2] [3] [4] [5] [6] [7]
- Completion of a DMA Transfer: [8]
- Dummy or Null Completion: [9]
- Region Overview: [10]
- Event Dataflow: [11] [12]
- Trigger Source Priority: [13] [14] [15] [16]
- Transfer Chaining Examples: [17] [18]
- EDMA Register Summary: [19] [20] [21] [22] [23] [24]
- EDMA Register Description: [25] [26] [27] [28] [29] [30] [31] [32] [33] [34] [35] [36] [37] [38] [39] [40] [41] [42] [43] [44]

Table 16-209. EDMA_TPCC_CERH

Address Offset	0x0000 101C		
Physical Address	0x4330 101C 0x40D1 101C 0x4151 101C 0x01D1 101C 0x420A 101C 0x421A 101C 0x422A 101C 0x423A 101C	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Chained Event Register (High Part): If EDMA_TPCC_CERH.En bit is set (regardless of state of EDMA_TPCC_EERH.En), then the corresponding DMA channel is prioritized vs. other pending DMA events for submission to the TC. EDMA_TPCC_CERH.En bit is set when a chaining completion code is returned from one of the 3PTCs via the completion interface, or is generated internally via Early Completion path. EDMA_TPCC_CERH.En bit is cleared when the corresponding event is prioritized and serviced. If the EDMA_TPCC_CERH.En bit is already set and the corresponding chaining completion code is returned from the TC, then the corresponding bit in the Event Missed Register is set. EDMA_TPCC_CERH.En cannot be set or cleared via software.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E63	E62	E61	E60	E59	E58	E57	E56	E55	E54	E53	E52	E51	E50	E49	E48	E47	E46	E45	E44	E43	E42	E41	E40	E39	E38	E37	E36	E35	E34	E33	E32

Bits	Field Name	Description	Type	Reset
31	E63	Event #63	R	0x0
30	E62	Event #62	R	0x0
29	E61	Event #61	R	0x0
28	E60	Event #60	R	0x0
27	E59	Event #59	R	0x0
26	E58	Event #58	R	0x0
25	E57	Event #57	R	0x0
24	E56	Event #56	R	0x0
23	E55	Event #55	R	0x0
22	E54	Event #54	R	0x0
21	E53	Event #53	R	0x0
20	E52	Event #52	R	0x0
19	E51	Event #51	R	0x0
18	E50	Event #50	R	0x0
17	E49	Event #49	R	0x0
16	E48	Event #48	R	0x0
15	E47	Event #47	R	0x0
14	E46	Event #46	R	0x0
13	E45	Event #45	R	0x0
12	E44	Event #44	R	0x0
11	E43	Event #43	R	0x0
10	E42	Event #42	R	0x0
9	E41	Event #41	R	0x0
8	E40	Event #40	R	0x0
7	E39	Event #39	R	0x0
6	E38	Event #38	R	0x0
5	E37	Event #37	R	0x0
4	E36	Event #36	R	0x0
3	E35	Event #35	R	0x0
2	E34	Event #34	R	0x0
1	E33	Event #33	R	0x0
0	E32	Event #32	R	0x0

Table 16-210. Register Call Summary for Register EDMA_TPCC_CERH

Enhanced DMA

- Dummy Versus Null Transfer Comparison: [0]
- Dummy or Null Completion: [1]
- Region Overview: [2]
- Event Dataflow: [3] [4]
- EDMA Register Summary: [5] [6] [7] [8] [9] [10]
- EDMA Register Description: [11] [12] [13] [14] [15] [16] [17] [18] [19] [20] [21] [22]

Table 16-211. EDMA_TPCC_EER

Address Offset	0x0000 1020		
Physical Address	0x4330 1020 0x40D1 1020 0x4151 1020 0x01D1 1020 0x420A 1020 0x421A 1020 0x422A 1020 0x423A 1020	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Event Enable Register: Enables DMA transfers for EDMA_TPCC_ER.En pending events. EDMA_TPCC_ER.En is set based on externally asserted events (via tpcc_eventN_pi). This register has no effect on Chained Event Register (EDMA_TPCC_CER) or Event Set Register (EDMA_TPCC_ESR). Note that if a bit is set in EDMA_TPCC_ER.En while EDMA_TPCC_EER.En is disabled, no action is taken. If EDMA_TPCC_EER.En is enabled at a later point (and EDMA_TPCC_ER.En has not been cleared via SW) then the event will be recognized as a valid 'TR Sync' EDMA_TPCC_EER.En is not directly writeable. Events can be enabled via writes to EDMA_TPCC_EESR and can be disabled via writes to EDMA_TPCC_EECR register. EDMA_TPCC_EER.En = 0: EDMA_TPCC_ER.En is not enabled to trigger DMA transfers. EDMA_TPCC_EER.En = 1: EDMA_TPCC_ER.En is enabled to trigger DMA transfers.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E31	E30	E29	E28	E27	E26	E25	E24	E23	E22	E21	E20	E19	E18	E17	E16	E15	E14	E13	E12	E11	E10	E9	E8	E7	E6	E5	E4	E3	E2	E1	E0

Bits	Field Name	Description	Type	Reset
31	E31	Event #31	R	0x0
30	E30	Event #30	R	0x0
29	E29	Event #29	R	0x0
28	E28	Event #28	R	0x0
27	E27	Event #27	R	0x0
26	E26	Event #26	R	0x0
25	E25	Event #25	R	0x0
24	E24	Event #24	R	0x0
23	E23	Event #23	R	0x0
22	E22	Event #22	R	0x0
21	E21	Event #21	R	0x0
20	E20	Event #20	R	0x0
19	E19	Event #19	R	0x0
18	E18	Event #18	R	0x0
17	E17	Event #17	R	0x0
16	E16	Event #16	R	0x0
15	E15	Event #15	R	0x0
14	E14	Event #14	R	0x0
13	E13	Event #13	R	0x0
12	E12	Event #12	R	0x0
11	E11	Event #11	R	0x0
10	E10	Event #10	R	0x0
9	E9	Event #9	R	0x0
8	E8	Event #8	R	0x0
7	E7	Event #7	R	0x0
6	E6	Event #6	R	0x0
5	E5	Event #5	R	0x0
4	E4	Event #4	R	0x0

Bits	Field Name	Description	Type	Reset
3	E3	Event #3	R	0x0
2	E2	Event #2	R	0x0
1	E1	Event #1	R	0x0
0	E0	Event #0	R	0x0

Table 16-212. Register Call Summary for Register EDMA_TPCC_EER

Enhanced DMA

- DMA Channel: [0] [1] [2] [3] [4] [5]
- Region Overview: [6] [7]
- Active Memory Protection: [8] [9] [10] [11] [12] [13] [14] [15] [16] [17] [18] [19] [20] [21]
- Peripheral Servicing Example: [22]
- Setting Up an EDMA Transfer: [23]
- EDMA Debug Checklist: [24]
- EDMA Programming Tips: [25]
- EDMA Register Summary: [26] [27] [28] [29] [30] [31]
- EDMA Register Description: [32] [33] [34] [35] [36] [37] [38] [39] [40] [41] [42] [43] [44] [45] [46] [47] [48] [49] [50] [51] [52] [53] [54] [55]

Table 16-213. EDMA_TPCC_EERH

Address Offset	0x0000 1024		
Physical Address	0x4330 1024 0x40D1 1024 0x4151 1024 0x01D1 1024 0x420A 1024 0x421A 1024 0x422A 1024 0x423A 1024	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Event Enable Register (High Part): Enables DMA transfers for EDMA_TPCC_ERH.En pending events. EDMA_TPCC_ERH.En is set based on externally asserted events (via tpcc_eventN_pi). This register has no effect on Chained Event Register (EDMA_TPCC_CERH) or Event Set Register (EDMA_TPCC_ESRH). Note that if a bit is set in EDMA_TPCC_ERH.En while EDMA_TPCC_EERH.En is disabled, no action is taken. If EDMA_TPCC_EERH.En is enabled at a later point (and EDMA_TPCC_ERH.En has not been cleared via SW) then the event will be recognized as a valid 'TR Sync' EDMA_TPCC_EERH.En is not directly writeable. Events can be enabled via writes to EDMA_TPCC_EESRH and can be disabled via writes to EDMA_TPCC_EECRH register. EDMA_TPCC_EERH.En = 0: EDMA_TPCC_ER.En is not enabled to trigger DMA transfers. EDMA_TPCC_EERH.En = 1: EDMA_TPCC_ER.En is enabled to trigger DMA transfers.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E63	E62	E61	E60	E59	E58	E57	E56	E55	E54	E53	E52	E51	E50	E49	E48	E47	E46	E45	E44	E43	E42	E41	E40	E39	E38	E37	E36	E35	E34	E33	E32

Bits	Field Name	Description	Type	Reset
31	E63	Event #63	R	0x0
30	E62	Event #62	R	0x0
29	E61	Event #61	R	0x0
28	E60	Event #60	R	0x0
27	E59	Event #59	R	0x0
26	E58	Event #58	R	0x0
25	E57	Event #57	R	0x0
24	E56	Event #56	R	0x0
23	E55	Event #55	R	0x0

Bits	Field Name	Description	Type	Reset
22	E54	Event #54	R	0x0
21	E53	Event #53	R	0x0
20	E52	Event #52	R	0x0
19	E51	Event #51	R	0x0
18	E50	Event #50	R	0x0
17	E49	Event #49	R	0x0
16	E48	Event #48	R	0x0
15	E47	Event #47	R	0x0
14	E46	Event #46	R	0x0
13	E45	Event #45	R	0x0
12	E44	Event #44	R	0x0
11	E43	Event #43	R	0x0
10	E42	Event #42	R	0x0
9	E41	Event #41	R	0x0
8	E40	Event #40	R	0x0
7	E39	Event #39	R	0x0
6	E38	Event #38	R	0x0
5	E37	Event #37	R	0x0
4	E36	Event #36	R	0x0
3	E35	Event #35	R	0x0
2	E34	Event #34	R	0x0
1	E33	Event #33	R	0x0
0	E32	Event #32	R	0x0

Table 16-214. Register Call Summary for Register EDMA_TPCC_EERH

Enhanced DMA

- Region Overview: [0]
- Setting Up an EDMA Transfer: [1]
- EDMA Debug Checklist: [2]
- EDMA Programming Tips: [3]
- EDMA Register Summary: [4] [5] [6] [7] [8] [9]
- EDMA Register Description: [10] [11] [12] [13] [14] [15] [16] [17] [18] [19] [20] [21] [22] [23] [24] [25] [26] [27] [28] [29]

Table 16-215. EDMA_TPCC_EECR

Address Offset	0x0000 1028		
Physical Address	0x4330 1028 0x40D1 1028 0x4151 1028 0x01D1 1028 0x420A 1028 0x421A 1028 0x422A 1028 0x423A 1028	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Event Enable Clear Register CPU writes of '1' to the EDMA_TPCC_EECR.En bit causes the EDMA_TPCC_EER.En bit to be cleared. CPU writes of '0' has no effect..		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E31	E30	E29	E28	E27	E26	E25	E24	E23	E22	E21	E20	E19	E18	E17	E16	E15	E14	E13	E12	E11	E10	E9	E8	E7	E6	E5	E4	E3	E2	E1	E0

Bits	Field Name	Description	Type	Reset
31	E31	Event #31	W	0x0
30	E30	Event #30	W	0x0
29	E29	Event #29	W	0x0
28	E28	Event #28	W	0x0
27	E27	Event #27	W	0x0
26	E26	Event #26	W	0x0
25	E25	Event #25	W	0x0
24	E24	Event #24	W	0x0
23	E23	Event #23	W	0x0
22	E22	Event #22	W	0x0
21	E21	Event #21	W	0x0
20	E20	Event #20	W	0x0
19	E19	Event #19	W	0x0
18	E18	Event #18	W	0x0
17	E17	Event #17	W	0x0
16	E16	Event #16	W	0x0
15	E15	Event #15	W	0x0
14	E14	Event #14	W	0x0
13	E13	Event #13	W	0x0
12	E12	Event #12	W	0x0
11	E11	Event #11	W	0x0
10	E10	Event #10	W	0x0
9	E9	Event #9	W	0x0
8	E8	Event #8	W	0x0
7	E7	Event #7	W	0x0
6	E6	Event #6	W	0x0
5	E5	Event #5	W	0x0
4	E4	Event #4	W	0x0
3	E3	Event #3	W	0x0
2	E2	Event #2	W	0x0
1	E1	Event #1	W	0x0
0	E0	Event #0	W	0x0

Table 16-216. Register Call Summary for Register EDMA_TPCC_EECR

Enhanced DMA

- Region Overview: [0]
- EDMA Programming Tips: [1]
- EDMA Register Summary: [2] [3] [4] [5] [6] [7]
- EDMA Register Description: [8] [9] [10] [11]

Table 16-217. EDMA_TPCC_EECRH

Address Offset	0x0000 102C		
Physical Address	0x4330 102C 0x40D1 102C 0x4151 102C 0x01D1 102C 0x420A 102C 0x421A 102C 0x422A 102C 0x423A 102C	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC

Table 16-217. EDMA_TPCC_EECRH (continued)

Description	Event Enable Clear Register (High Part) CPU writes of '1' to the EDMA_TPCC_EECRH.En bit causes the EERH.En bit to be cleared. CPU writes of '0' has no effect..
Type	W

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E63	E62	E61	E60	E59	E58	E57	E56	E55	E54	E53	E52	E51	E50	E49	E48	E47	E46	E45	E44	E43	E42	E41	E40	E39	E38	E37	E36	E35	E34	E33	E32

Bits	Field Name	Description	Type	Reset
31	E63	Event #63	W	0x0
30	E62	Event #62	W	0x0
29	E61	Event #61	W	0x0
28	E60	Event #60	W	0x0
27	E59	Event #59	W	0x0
26	E58	Event #58	W	0x0
25	E57	Event #57	W	0x0
24	E56	Event #56	W	0x0
23	E55	Event #55	W	0x0
22	E54	Event #54	W	0x0
21	E53	Event #53	W	0x0
20	E52	Event #52	W	0x0
19	E51	Event #51	W	0x0
18	E50	Event #50	W	0x0
17	E49	Event #49	W	0x0
16	E48	Event #48	W	0x0
15	E47	Event #47	W	0x0
14	E46	Event #46	W	0x0
13	E45	Event #45	W	0x0
12	E44	Event #44	W	0x0
11	E43	Event #43	W	0x0
10	E42	Event #42	W	0x0
9	E41	Event #41	W	0x0
8	E40	Event #40	W	0x0
7	E39	Event #39	W	0x0
6	E38	Event #38	W	0x0
5	E37	Event #37	W	0x0
4	E36	Event #36	W	0x0
3	E35	Event #35	W	0x0
2	E34	Event #34	W	0x0
1	E33	Event #33	W	0x0
0	E32	Event #32	W	0x0

Table 16-218. Register Call Summary for Register EDMA_TPCC_EECRH

Enhanced DMA

- Region Overview: [0]
- EDMA Programming Tips: [1]
- EDMA Register Summary: [2] [3] [4] [5] [6] [7]
- EDMA Register Description: [8] [9] [10] [11]

Table 16-219. EDMA_TPCC_EESR

Address Offset	0x0000 1030		
Physical Address	0x4330 1030 0x40D1 1030 0x4151 1030 0x01D1 1030 0x420A 1030 0x421A 1030 0x422A 1030 0x423A 1030	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Event Enable Set Register CPU write of '1' to the EDMA_TPCC_EESR.En bit causes the EDMA_TPCC_EER.En bit to be set. CPU writes of '0' has no effect..		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E31	E30	E29	E28	E27	E26	E25	E24	E23	E22	E21	E20	E19	E18	E17	E16	E15	E14	E13	E12	E11	E10	E9	E8	E7	E6	E5	E4	E3	E2	E1	E0

Bits	Field Name	Description	Type	Reset
31	E31	Event #31	W	0x0
30	E30	Event #30	W	0x0
29	E29	Event #29	W	0x0
28	E28	Event #28	W	0x0
27	E27	Event #27	W	0x0
26	E26	Event #26	W	0x0
25	E25	Event #25	W	0x0
24	E24	Event #24	W	0x0
23	E23	Event #23	W	0x0
22	E22	Event #22	W	0x0
21	E21	Event #21	W	0x0
20	E20	Event #20	W	0x0
19	E19	Event #19	W	0x0
18	E18	Event #18	W	0x0
17	E17	Event #17	W	0x0
16	E16	Event #16	W	0x0
15	E15	Event #15	W	0x0
14	E14	Event #14	W	0x0
13	E13	Event #13	W	0x0
12	E12	Event #12	W	0x0
11	E11	Event #11	W	0x0
10	E10	Event #10	W	0x0
9	E9	Event #9	W	0x0
8	E8	Event #8	W	0x0
7	E7	Event #7	W	0x0
6	E6	Event #6	W	0x0
5	E5	Event #5	W	0x0
4	E4	Event #4	W	0x0
3	E3	Event #3	W	0x0
2	E2	Event #2	W	0x0
1	E1	Event #1	W	0x0
0	E0	Event #0	W	0x0

Table 16-220. Register Call Summary for Register EDMA_TPCC_EESR

Enhanced DMA

- Region Overview: [0]
- Active Memory Protection: [1] [2] [3] [4] [5] [6] [7] [8] [9] [10] [11] [12]
- Setting Up an EDMA Transfer: [13]
- EDMA Programming Tips: [14]
- EDMA Register Summary: [15] [16] [17] [18] [19] [20]
- EDMA Register Description: [21] [22] [23] [24]

Table 16-221. EDMA_TPCC_EESRH

Address Offset	0x0000 1034		
Physical Address	0x4330 1034 0x40D1 1034 0x4151 1034 0x01D1 1034 0x420A 1034 0x421A 1034 0x422A 1034 0x423A 1034	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Event Enable Set Register (High Part) CPU writes of '1' to the EDMA_TPCC_EESRH.En bit causes the EDMA_TPCC_EERH.En bit to be set. CPU writes of '0' has no effect..		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E63	E62	E61	E60	E59	E58	E57	E56	E55	E54	E53	E52	E51	E50	E49	E48	E47	E46	E45	E44	E43	E42	E41	E40	E39	E38	E37	E36	E35	E34	E33	E32

Bits	Field Name	Description	Type	Reset
31	E63	Event #63	W	0x0
30	E62	Event #62	W	0x0
29	E61	Event #61	W	0x0
28	E60	Event #60	W	0x0
27	E59	Event #59	W	0x0
26	E58	Event #58	W	0x0
25	E57	Event #57	W	0x0
24	E56	Event #56	W	0x0
23	E55	Event #55	W	0x0
22	E54	Event #54	W	0x0
21	E53	Event #53	W	0x0
20	E52	Event #52	W	0x0
19	E51	Event #51	W	0x0
18	E50	Event #50	W	0x0
17	E49	Event #49	W	0x0
16	E48	Event #48	W	0x0
15	E47	Event #47	W	0x0
14	E46	Event #46	W	0x0
13	E45	Event #45	W	0x0
12	E44	Event #44	W	0x0
11	E43	Event #43	W	0x0
10	E42	Event #42	W	0x0
9	E41	Event #41	W	0x0

Bits	Field Name	Description	Type	Reset
8	E40	Event #40	W	0x0
7	E39	Event #39	W	0x0
6	E38	Event #38	W	0x0
5	E37	Event #37	W	0x0
4	E36	Event #36	W	0x0
3	E35	Event #35	W	0x0
2	E34	Event #34	W	0x0
1	E33	Event #33	W	0x0
0	E32	Event #32	W	0x0

Table 16-222. Register Call Summary for Register EDMA_TPCC_EESRH

Enhanced DMA

- Region Overview: [0]
- Setting Up an EDMA Transfer: [1]
- EDMA Programming Tips: [2]
- EDMA Register Summary: [3] [4] [5] [6] [7] [8]
- EDMA Register Description: [9] [10] [11] [12]

Table 16-223. EDMA_TPCC_SER

Address Offset	0x0000 1038		
Physical Address	0x4330 1038 0x40D1 1038 0x4151 1038 0x01D1 1038 0x420A 1038 0x421A 1038 0x422A 1038 0x423A 1038	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Secondary Event Register The secondary event register is used along with the Event Register (EDMA_TPCC_ER) to provide information on the state of an Event. En = 0 : Event is not currently in the Event Queue. En = 1 : Event is currently stored in Event Queue. Event arbiter will not prioritize additional events.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E31	E30	E29	E28	E27	E26	E25	E24	E23	E22	E21	E20	E19	E18	E17	E16	E15	E14	E13	E12	E11	E10	E9	E8	E7	E6	E5	E4	E3	E2	E1	E0

Bits	Field Name	Description	Type	Reset
31	E31	Event #31	R	0x0
30	E30	Event #30	R	0x0
29	E29	Event #29	R	0x0
28	E28	Event #28	R	0x0
27	E27	Event #27	R	0x0
26	E26	Event #26	R	0x0
25	E25	Event #25	R	0x0
24	E24	Event #24	R	0x0
23	E23	Event #23	R	0x0
22	E22	Event #22	R	0x0
21	E21	Event #21	R	0x0

Bits	Field Name	Description	Type	Reset
20	E20	Event #20	R	0x0
19	E19	Event #19	R	0x0
18	E18	Event #18	R	0x0
17	E17	Event #17	R	0x0
16	E16	Event #16	R	0x0
15	E15	Event #15	R	0x0
14	E14	Event #14	R	0x0
13	E13	Event #13	R	0x0
12	E12	Event #12	R	0x0
11	E11	Event #11	R	0x0
10	E10	Event #10	R	0x0
9	E9	Event #9	R	0x0
8	E8	Event #8	R	0x0
7	E7	Event #7	R	0x0
6	E6	Event #6	R	0x0
5	E5	Event #5	R	0x0
4	E4	Event #4	R	0x0
3	E3	Event #3	R	0x0
2	E2	Event #2	R	0x0
1	E1	Event #1	R	0x0
0	E0	Event #0	R	0x0

Table 16-224. Register Call Summary for Register EDMA_TPCC_SER

Enhanced DMA

- Null PaRAM Set: [0] [1]
- Dummy PaRAM Set: [2]
- Dummy Versus Null Transfer Comparison: [3] [4]
- Region Overview: [5]
- Event Dataflow: [6] [7] [8]
- EDMA Debug Checklist: [9] [10] [11]
- EDMA Register Summary: [12] [13] [14] [15] [16] [17]
- EDMA Register Description: [18] [19] [20] [21] [22] [23]

Table 16-225. EDMA_TPCC_SERH

Address Offset	0x0000 103C		
Physical Address	0x4330 103C 0x40D1 103C 0x4151 103C 0x01D1 103C 0x420A 103C 0x421A 103C 0x422A 103C 0x423A 103C	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Secondary Event Register (High Part) The secondary event register is used along with the Event Register (EDMA_TPCC_ERH) to provide information on the state of an Event. En = 0 : Event is not currently in the Event Queue. En = 1 : Event is currently stored in Event Queue. Event arbiter will not prioritize additional events.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E63	E62	E61	E60	E59	E58	E57	E56	E55	E54	E53	E52	E51	E50	E49	E48	E47	E46	E45	E44	E43	E42	E41	E40	E39	E38	E37	E36	E35	E34	E33	E32

Bits	Field Name	Description	Type	Reset
31	E63	Event #63	R	0x0
30	E62	Event #62	R	0x0
29	E61	Event #61	R	0x0
28	E60	Event #60	R	0x0
27	E59	Event #59	R	0x0
26	E58	Event #58	R	0x0
25	E57	Event #57	R	0x0
24	E56	Event #56	R	0x0
23	E55	Event #55	R	0x0
22	E54	Event #54	R	0x0
21	E53	Event #53	R	0x0
20	E52	Event #52	R	0x0
19	E51	Event #51	R	0x0
18	E50	Event #50	R	0x0
17	E49	Event #49	R	0x0
16	E48	Event #48	R	0x0
15	E47	Event #47	R	0x0
14	E46	Event #46	R	0x0
13	E45	Event #45	R	0x0
12	E44	Event #44	R	0x0
11	E43	Event #43	R	0x0
10	E42	Event #42	R	0x0
9	E41	Event #41	R	0x0
8	E40	Event #40	R	0x0
7	E39	Event #39	R	0x0
6	E38	Event #38	R	0x0
5	E37	Event #37	R	0x0
4	E36	Event #36	R	0x0
3	E35	Event #35	R	0x0
2	E34	Event #34	R	0x0
1	E33	Event #33	R	0x0
0	E32	Event #32	R	0x0

Table 16-226. Register Call Summary for Register EDMA_TPCC_SERH

Enhanced DMA

- Null PaRAM Set: [0] [1]
- Dummy PaRAM Set: [2]
- Dummy Versus Null Transfer Comparison: [3]
- Region Overview: [4]
- Event Dataflow: [5] [6] [7]
- EDMA Debug Checklist: [8] [9]
- EDMA Register Summary: [10] [11] [12] [13] [14] [15]
- EDMA Register Description: [16] [17] [18] [19]

Table 16-227. EDMA_TPCC_SECR

Address Offset	0x0000 1040		
Physical Address	0x4330 1040 0x40D1 1040 0x4151 1040 0x01D1 1040 0x420A 1040 0x421A 1040 0x422A 1040 0x423A 1040	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Secondary Event Clear Register The secondary event clear register is used to clear the status of the EDMA_TPCC_SER registers. CPU write of '1' to the EDMA_TPCC_SECR.En bit clears the EDMA_TPCC_SER register. CPU write of '0' has no effect.		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E31	E30	E29	E28	E27	E26	E25	E24	E23	E22	E21	E20	E19	E18	E17	E16	E15	E14	E13	E12	E11	E10	E9	E8	E7	E6	E5	E4	E3	E2	E1	E0

Bits	Field Name	Description	Type	Reset
31	E31	Event #31	W	0x0
30	E30	Event #30	W	0x0
29	E29	Event #29	W	0x0
28	E28	Event #28	W	0x0
27	E27	Event #27	W	0x0
26	E26	Event #26	W	0x0
25	E25	Event #25	W	0x0
24	E24	Event #24	W	0x0
23	E23	Event #23	W	0x0
22	E22	Event #22	W	0x0
21	E21	Event #21	W	0x0
20	E20	Event #20	W	0x0
19	E19	Event #19	W	0x0
18	E18	Event #18	W	0x0
17	E17	Event #17	W	0x0
16	E16	Event #16	W	0x0
15	E15	Event #15	W	0x0
14	E14	Event #14	W	0x0
13	E13	Event #13	W	0x0
12	E12	Event #12	W	0x0
11	E11	Event #11	W	0x0
10	E10	Event #10	W	0x0
9	E9	Event #9	W	0x0
8	E8	Event #8	W	0x0
7	E7	Event #7	W	0x0
6	E6	Event #6	W	0x0
5	E5	Event #5	W	0x0
4	E4	Event #4	W	0x0
3	E3	Event #3	W	0x0
2	E2	Event #2	W	0x0
1	E1	Event #1	W	0x0
0	E0	Event #0	W	0x0

Table 16-228. Register Call Summary for Register EDMA_TPCC_SECR

Enhanced DMA
- Region Overview: [0]
- EDMA Register Summary: [1] [2] [3] [4] [5] [6]
- EDMA Register Description: [7] [8]

Table 16-229. EDMA_TPCC_SECRH

Address Offset	0x0000 1044		
Physical Address	0x4330 1044 0x40D1 1044 0x4151 1044 0x01D1 1044 0x420A 1044 0x421A 1044 0x422A 1044 0x423A 1044	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Secondary Event Clear Register (High Part) The secondary event clear register is used to clear the status of the EDMA_TPCC_SERH registers. CPU write of '1' to the EDMA_TPCC_SECRH.En bit clears the EDMA_TPCC_SERH register. CPU write of '0' has no effect.		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E63	E62	E61	E60	E59	E58	E57	E56	E55	E54	E53	E52	E51	E50	E49	E48	E47	E46	E45	E44	E43	E42	E41	E40	E39	E38	E37	E36	E35	E34	E33	E32

Bits	Field Name	Description	Type	Reset
31	E63	Event #63	W	0x0
30	E62	Event #62	W	0x0
29	E61	Event #61	W	0x0
28	E60	Event #60	W	0x0
27	E59	Event #59	W	0x0
26	E58	Event #58	W	0x0
25	E57	Event #57	W	0x0
24	E56	Event #56	W	0x0
23	E55	Event #55	W	0x0
22	E54	Event #54	W	0x0
21	E53	Event #53	W	0x0
20	E52	Event #52	W	0x0
19	E51	Event #51	W	0x0
18	E50	Event #50	W	0x0
17	E49	Event #49	W	0x0
16	E48	Event #48	W	0x0
15	E47	Event #47	W	0x0
14	E46	Event #46	W	0x0
13	E45	Event #45	W	0x0
12	E44	Event #44	W	0x0
11	E43	Event #43	W	0x0
10	E42	Event #42	W	0x0
9	E41	Event #41	W	0x0
8	E40	Event #40	W	0x0
7	E39	Event #39	W	0x0

Bits	Field Name	Description	Type	Reset
6	E38	Event #38	W	0x0
5	E37	Event #37	W	0x0
4	E36	Event #36	W	0x0
3	E35	Event #35	W	0x0
2	E34	Event #34	W	0x0
1	E33	Event #33	W	0x0
0	E32	Event #32	W	0x0

Table 16-230. Register Call Summary for Register EDMA_TPCC_SECRH

Enhanced DMA
- Region Overview: [0]
- EDMA Register Summary: [1] [2] [3] [4] [5] [6]
- EDMA Register Description: [7] [8]

Table 16-231. EDMA_TPCC_IER

Address Offset	0x0000 1050		
Physical Address	0x4330 1050 0x40D1 1050 0x4151 1050 0x01D1 1050 0x420A 1050 0x421A 1050 0x422A 1050 0x423A 1050	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Int Enable Register EDMA_TPCC_IER.In is not directly writeable. Interrupts can be enabled via writes to EDMA_TPCC_IESR and can be disabled via writes to EDMA_TPCC_IECR register. EDMA_TPCC_IER. In = 0: EDMA_TPCC_IPR.In is NOT enabled for interrupts. EDMA_TPCC_IER. In = 1: EDMA_TPCC_IPR.In IS enabled for interrupts.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
I31	I30	I29	I28	I27	I26	I25	I24	I23	I22	I21	I20	I19	I18	I17	I16	I15	I14	I13	I12	I11	I10	I9	I8	I7	I6	I5	I4	I3	I2	I1	I0

Bits	Field Name	Description	Type	Reset
31	I31	Interrupt associated with TCC #31	R	0x0
30	I30	Interrupt associated with TCC #30	R	0x0
29	I29	Interrupt associated with TCC #29	R	0x0
28	I28	Interrupt associated with TCC #28	R	0x0
27	I27	Interrupt associated with TCC #27	R	0x0
26	I26	Interrupt associated with TCC #26	R	0x0
25	I25	Interrupt associated with TCC #25	R	0x0
24	I24	Interrupt associated with TCC #24	R	0x0
23	I23	Interrupt associated with TCC #23	R	0x0
22	I22	Interrupt associated with TCC #22	R	0x0
21	I21	Interrupt associated with TCC #21	R	0x0
20	I20	Interrupt associated with TCC #20	R	0x0
19	I19	Interrupt associated with TCC #19	R	0x0
18	I18	Interrupt associated with TCC #18	R	0x0
17	I17	Interrupt associated with TCC #17	R	0x0

Bits	Field Name	Description	Type	Reset
16	I16	Interrupt associated with TCC #16	R	0x0
15	I15	Interrupt associated with TCC #15	R	0x0
14	I14	Interrupt associated with TCC #14	R	0x0
13	I13	Interrupt associated with TCC #13	R	0x0
12	I12	Interrupt associated with TCC #12	R	0x0
11	I11	Interrupt associated with TCC #11	R	0x0
10	I10	Interrupt associated with TCC #10	R	0x0
9	I9	Interrupt associated with TCC #9	R	0x0
8	I8	Interrupt associated with TCC #8	R	0x0
7	I7	Interrupt associated with TCC #7	R	0x0
6	I6	Interrupt associated with TCC #6	R	0x0
5	I5	Interrupt associated with TCC #5	R	0x0
4	I4	Interrupt associated with TCC #4	R	0x0
3	I3	Interrupt associated with TCC #3	R	0x0
2	I2	Interrupt associated with TCC #2	R	0x0
1	I1	Interrupt associated with TCC #1	R	0x0
0	I0	Interrupt associated with TCC #0	R	0x0

Table 16-232. Register Call Summary for Register EDMA_TPCC_IER

Enhanced DMA

- Region Overview: [0]
- Region Interrupts: [1]
- Transfer Completion Interrupts: [2] [3] [4] [5] [6] [7] [8] [9] [10] [11] [12] [13] [14] [15] [16]
- Interrupt Evaluation Operations: [17]
- Transfer Chaining Examples: [18]
- Setting Up an EDMA Transfer: [19]
- EDMA Debug Checklist: [20] [21]
- EDMA Register Summary: [22] [23] [24] [25] [26] [27]
- EDMA Register Description: [28] [29] [30] [31] [32] [33] [34] [35] [36] [37] [38] [39]

Table 16-233. EDMA_TPCC_IERH

Address Offset	0x0000 1054		
Physical Address	0x4330 1054 0x40D1 1054 0x4151 1054 0x01D1 1054 0x420A 1054 0x421A 1054 0x422A 1054 0x423A 1054	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Int Enable Register (High Part) EDMA_TPCC_IERH.In is not directly writeable. Interrupts can be enabled via writes to EDMA_TPCC_IESRH and can be disabled via writes to EDMA_TPCC_IECRH register. EDMA_TPCC_IERH. In = 0: EDMA_TPCC_IPRH.In is NOT enabled for interrupts. EDMA_TPCC_IERH. In = 1: EDMA_TPCC_IPRH.In IS enabled for interrupts.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
I63	I62	I61	I60	I59	I58	I57	I56	I55	I54	I53	I52	I51	I50	I49	I48	I47	I46	I45	I44	I43	I42	I41	I40	I39	I38	I37	I36	I35	I34	I33	I32

Bits	Field Name	Description	Type	Reset
31	I63	Interrupt associated with TCC #63	R	0x0
30	I62	Interrupt associated with TCC #62	R	0x0
29	I61	Interrupt associated with TCC #61	R	0x0
28	I60	Interrupt associated with TCC #60	R	0x0
27	I59	Interrupt associated with TCC #59	R	0x0
26	I58	Interrupt associated with TCC #58	R	0x0
25	I57	Interrupt associated with TCC #57	R	0x0
24	I56	Interrupt associated with TCC #56	R	0x0
23	I55	Interrupt associated with TCC #55	R	0x0
22	I54	Interrupt associated with TCC #54	R	0x0
21	I53	Interrupt associated with TCC #53	R	0x0
20	I52	Interrupt associated with TCC #52	R	0x0
19	I51	Interrupt associated with TCC #51	R	0x0
18	I50	Interrupt associated with TCC #50	R	0x0
17	I49	Interrupt associated with TCC #49	R	0x0
16	I48	Interrupt associated with TCC #48	R	0x0
15	I47	Interrupt associated with TCC #47	R	0x0
14	I46	Interrupt associated with TCC #46	R	0x0
13	I45	Interrupt associated with TCC #45	R	0x0
12	I44	Interrupt associated with TCC #44	R	0x0
11	I43	Interrupt associated with TCC #43	R	0x0
10	I42	Interrupt associated with TCC #42	R	0x0
9	I41	Interrupt associated with TCC #41	R	0x0
8	I40	Interrupt associated with TCC #40	R	0x0
7	I39	Interrupt associated with TCC #39	R	0x0
6	I38	Interrupt associated with TCC #38	R	0x0
5	I37	Interrupt associated with TCC #37	R	0x0
4	I36	Interrupt associated with TCC #36	R	0x0
3	I35	Interrupt associated with TCC #35	R	0x0
2	I34	Interrupt associated with TCC #34	R	0x0
1	I33	Interrupt associated with TCC #33	R	0x0
0	I32	Interrupt associated with TCC #32	R	0x0

Table 16-234. Register Call Summary for Register EDMA_TPCC_IERH

Enhanced DMA

- Region Overview: [0]
- Transfer Completion Interrupts: [1] [2] [3] [4] [5] [6] [7] [8] [9]
- Interrupt Evaluation Operations: [10]
- Setting Up an EDMA Transfer: [11]
- EDMA Debug Checklist: [12] [13]
- EDMA Register Summary: [14] [15] [16] [17] [18] [19]
- EDMA Register Description: [20] [21] [22] [23] [24] [25] [26] [27]

Table 16-235. EDMA_TPCC_IECR

Address Offset	0x0000 1058		
Physical Address	0x4330 1058 0x40D1 1058 0x4151 1058 0x01D1 1058 0x420A 1058 0x421A 1058 0x422A 1058 0x423A 1058	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Int Enable Clear Register CPU writes of '1' to the EDMA_TPCC_IECR.In bit causes the EDMA_TPCC_IER.In bit to be cleared. CPU writes of '0' has no effect..		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
I31	I30	I29	I28	I27	I26	I25	I24	I23	I22	I21	I20	I19	I18	I17	I16	I15	I14	I13	I12	I11	I10	I9	I8	I7	I6	I5	I4	I3	I2	I1	I0

Bits	Field Name	Description	Type	Reset
31	I31	Interrupt associated with TCC #31	W	0x0
30	I30	Interrupt associated with TCC #30	W	0x0
29	I29	Interrupt associated with TCC #29	W	0x0
28	I28	Interrupt associated with TCC #28	W	0x0
27	I27	Interrupt associated with TCC #27	W	0x0
26	I26	Interrupt associated with TCC #26	W	0x0
25	I25	Interrupt associated with TCC #25	W	0x0
24	I24	Interrupt associated with TCC #24	W	0x0
23	I23	Interrupt associated with TCC #23	W	0x0
22	I22	Interrupt associated with TCC #22	W	0x0
21	I21	Interrupt associated with TCC #21	W	0x0
20	I20	Interrupt associated with TCC #20	W	0x0
19	I19	Interrupt associated with TCC #19	W	0x0
18	I18	Interrupt associated with TCC #18	W	0x0
17	I17	Interrupt associated with TCC #17	W	0x0
16	I16	Interrupt associated with TCC #16	W	0x0
15	I15	Interrupt associated with TCC #15	W	0x0
14	I14	Interrupt associated with TCC #14	W	0x0
13	I13	Interrupt associated with TCC #13	W	0x0
12	I12	Interrupt associated with TCC #12	W	0x0
11	I11	Interrupt associated with TCC #11	W	0x0
10	I10	Interrupt associated with TCC #10	W	0x0
9	I9	Interrupt associated with TCC #9	W	0x0
8	I8	Interrupt associated with TCC #8	W	0x0
7	I7	Interrupt associated with TCC #7	W	0x0
6	I6	Interrupt associated with TCC #6	W	0x0
5	I5	Interrupt associated with TCC #5	W	0x0
4	I4	Interrupt associated with TCC #4	W	0x0
3	I3	Interrupt associated with TCC #3	W	0x0
2	I2	Interrupt associated with TCC #2	W	0x0
1	I1	Interrupt associated with TCC #1	W	0x0
0	I0	Interrupt associated with TCC #0	W	0x0

Table 16-236. Register Call Summary for Register EDMA_TPCC_IECR

Enhanced DMA
- Region Overview: [0]
- Transfer Completion Interrupts: [1]
- EDMA Register Summary: [2] [3] [4] [5] [6] [7]
- EDMA Register Description: [8] [9] [10] [11]

Table 16-237. EDMA_TPCC_IECRH

Address Offset	0x0000 105C		
Physical Address	0x4330 105C 0x40D1 105C 0x4151 105C 0x01D1 105C 0x420A 105C 0x421A 105C 0x422A 105C 0x423A 105C	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Int Enable Clear Register (High Part) CPU write of '1' to the EDMA_TPCC_IECRH.In bit causes the EDMA_TPCC_IERH.In bit to be cleared. CPU write of '0' has no effect..		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
I63	I62	I61	I60	I59	I58	I57	I56	I55	I54	I53	I52	I51	I50	I49	I48	I47	I46	I45	I44	I43	I42	I41	I40	I39	I38	I37	I36	I35	I34	I33	I32

Bits	Field Name	Description	Type	Reset
31	I63	Interrupt associated with TCC #63	W	0x0
30	I62	Interrupt associated with TCC #62	W	0x0
29	I61	Interrupt associated with TCC #61	W	0x0
28	I60	Interrupt associated with TCC #60	W	0x0
27	I59	Interrupt associated with TCC #59	W	0x0
26	I58	Interrupt associated with TCC #58	W	0x0
25	I57	Interrupt associated with TCC #57	W	0x0
24	I56	Interrupt associated with TCC #56	W	0x0
23	I55	Interrupt associated with TCC #55	W	0x0
22	I54	Interrupt associated with TCC #54	W	0x0
21	I53	Interrupt associated with TCC #53	W	0x0
20	I52	Interrupt associated with TCC #52	W	0x0
19	I51	Interrupt associated with TCC #51	W	0x0
18	I50	Interrupt associated with TCC #50	W	0x0
17	I49	Interrupt associated with TCC #49	W	0x0
16	I48	Interrupt associated with TCC #48	W	0x0
15	I47	Interrupt associated with TCC #47	W	0x0
14	I46	Interrupt associated with TCC #46	W	0x0
13	I45	Interrupt associated with TCC #45	W	0x0
12	I44	Interrupt associated with TCC #44	W	0x0
11	I43	Interrupt associated with TCC #43	W	0x0
10	I42	Interrupt associated with TCC #42	W	0x0
9	I41	Interrupt associated with TCC #41	W	0x0
8	I40	Interrupt associated with TCC #40	W	0x0
7	I39	Interrupt associated with TCC #39	W	0x0

Bits	Field Name	Description	Type	Reset
6	I38	Interrupt associated with TCC #38	W	0x0
5	I37	Interrupt associated with TCC #37	W	0x0
4	I36	Interrupt associated with TCC #36	W	0x0
3	I35	Interrupt associated with TCC #35	W	0x0
2	I34	Interrupt associated with TCC #34	W	0x0
1	I33	Interrupt associated with TCC #33	W	0x0
0	I32	Interrupt associated with TCC #32	W	0x0

Table 16-238. Register Call Summary for Register EDMA_TPCC_IECRH

Enhanced DMA

- Region Overview: [0]
- EDMA Register Summary: [1] [2] [3] [4] [5] [6]
- EDMA Register Description: [7] [8] [9] [10]

Table 16-239. EDMA_TPCC_IESR

Address Offset	0x0000 1060		
Physical Address	0x4330 1060 0x40D1 1060 0x4151 1060 0x01D1 1060 0x420A 1060 0x421A 1060 0x422A 1060 0x423A 1060	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Int Enable Set Register CPU write of '1' to the EDMA_TPCC_IESR.In bit causes the EDMA_TPCC_IESR.In bit to be set. CPU write of '0' has no effect..		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
I31	I30	I29	I28	I27	I26	I25	I24	I23	I22	I21	I20	I19	I18	I17	I16	I15	I14	I13	I12	I11	I10	I9	I8	I7	I6	I5	I4	I3	I2	I1	I0

Bits	Field Name	Description	Type	Reset
31	I31	Interrupt associated with TCC #31	W	0x0
30	I30	Interrupt associated with TCC #30	W	0x0
29	I29	Interrupt associated with TCC #29	W	0x0
28	I28	Interrupt associated with TCC #28	W	0x0
27	I27	Interrupt associated with TCC #27	W	0x0
26	I26	Interrupt associated with TCC #26	W	0x0
25	I25	Interrupt associated with TCC #25	W	0x0
24	I24	Interrupt associated with TCC #24	W	0x0
23	I23	Interrupt associated with TCC #23	W	0x0
22	I22	Interrupt associated with TCC #22	W	0x0
21	I21	Interrupt associated with TCC #21	W	0x0
20	I20	Interrupt associated with TCC #20	W	0x0
19	I19	Interrupt associated with TCC #19	W	0x0
18	I18	Interrupt associated with TCC #18	W	0x0
17	I17	Interrupt associated with TCC #17	W	0x0
16	I16	Interrupt associated with TCC #16	W	0x0
15	I15	Interrupt associated with TCC #15	W	0x0

Bits	Field Name	Description	Type	Reset
14	I14	Interrupt associated with TCC #14	W	0x0
13	I13	Interrupt associated with TCC #13	W	0x0
12	I12	Interrupt associated with TCC #12	W	0x0
11	I11	Interrupt associated with TCC #11	W	0x0
10	I10	Interrupt associated with TCC #10	W	0x0
9	I9	Interrupt associated with TCC #9	W	0x0
8	I8	Interrupt associated with TCC #8	W	0x0
7	I7	Interrupt associated with TCC #7	W	0x0
6	I6	Interrupt associated with TCC #6	W	0x0
5	I5	Interrupt associated with TCC #5	W	0x0
4	I4	Interrupt associated with TCC #4	W	0x0
3	I3	Interrupt associated with TCC #3	W	0x0
2	I2	Interrupt associated with TCC #2	W	0x0
1	I1	Interrupt associated with TCC #1	W	0x0
0	I0	Interrupt associated with TCC #0	W	0x0

Table 16-240. Register Call Summary for Register EDMA_TPCC_IESR

Enhanced DMA

- Region Overview: [0]
- Transfer Completion Interrupts: [1]
- Setting Up an EDMA Transfer: [2]
- EDMA Register Summary: [3] [4] [5] [6] [7] [8]
- EDMA Register Description: [9] [10] [11] [12] [13] [14]

Table 16-241. EDMA_TPCC_IESRH

Address Offset	0x0000 1064		
Physical Address	0x4330 1064 0x40D1 1064 0x4151 1064 0x01D1 1064 0x420A 1064 0x421A 1064 0x422A 1064 0x423A 1064	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Int Enable Set Register (High Part) CPU write of '1' to the EDMA_TPCC_IESRH.In bit causes the EDMA_TPCC_IESRH.In bit to be set. CPU write of '0' has no effect..		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
I63	I62	I61	I60	I59	I58	I57	I56	I55	I54	I53	I52	I51	I50	I49	I48	I47	I46	I45	I44	I43	I42	I41	I40	I39	I38	I37	I36	I35	I34	I33	I32

Bits	Field Name	Description	Type	Reset
31	I63	Interrupt associated with TCC #63	W	0x0
30	I62	Interrupt associated with TCC #62	W	0x0
29	I61	Interrupt associated with TCC #61	W	0x0
28	I60	Interrupt associated with TCC #60	W	0x0
27	I59	Interrupt associated with TCC #59	W	0x0
26	I58	Interrupt associated with TCC #58	W	0x0
25	I57	Interrupt associated with TCC #57	W	0x0

Bits	Field Name	Description	Type	Reset
24	I56	Interrupt associated with TCC #56	W	0x0
23	I55	Interrupt associated with TCC #55	W	0x0
22	I54	Interrupt associated with TCC #54	W	0x0
21	I53	Interrupt associated with TCC #53	W	0x0
20	I52	Interrupt associated with TCC #52	W	0x0
19	I51	Interrupt associated with TCC #51	W	0x0
18	I50	Interrupt associated with TCC #50	W	0x0
17	I49	Interrupt associated with TCC #49	W	0x0
16	I48	Interrupt associated with TCC #48	W	0x0
15	I47	Interrupt associated with TCC #47	W	0x0
14	I46	Interrupt associated with TCC #46	W	0x0
13	I45	Interrupt associated with TCC #45	W	0x0
12	I44	Interrupt associated with TCC #44	W	0x0
11	I43	Interrupt associated with TCC #43	W	0x0
10	I42	Interrupt associated with TCC #42	W	0x0
9	I41	Interrupt associated with TCC #41	W	0x0
8	I40	Interrupt associated with TCC #40	W	0x0
7	I39	Interrupt associated with TCC #39	W	0x0
6	I38	Interrupt associated with TCC #38	W	0x0
5	I37	Interrupt associated with TCC #37	W	0x0
4	I36	Interrupt associated with TCC #36	W	0x0
3	I35	Interrupt associated with TCC #35	W	0x0
2	I34	Interrupt associated with TCC #34	W	0x0
1	I33	Interrupt associated with TCC #33	W	0x0
0	I32	Interrupt associated with TCC #32	W	0x0

Table 16-242. Register Call Summary for Register EDMA_TPCC_IESRH

Enhanced DMA

- Region Overview: [0]
- Setting Up an EDMA Transfer: [1]
- EDMA Register Summary: [2] [3] [4] [5] [6] [7]
- EDMA Register Description: [8] [9] [10] [11] [12] [13]

Table 16-243. EDMA_TPCC_IPR

Address Offset	0x0000 1068		
Physical Address	0x4330 1068 0x40D1 1068 0x4151 1068 0x01D1 1068 0x420A 1068 0x421A 1068 0x422A 1068 0x423A 1068	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Interrupt Pending Register EDMA_TPCC_IPR.In bit is set when a interrupt completion code with TCC of N is detected. EDMA_TPCC_IPR.In bit is cleared via software by writing a '1' to EDMA_TPCC_ICR.In bit.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
I31	I30	I29	I28	I27	I26	I25	I24	I23	I22	I21	I20	I19	I18	I17	I16	I15	I14	I13	I12	I11	I10	I9	I8	I7	I6	I5	I4	I3	I2	I1	I0

Bits	Field Name	Description	Type	Reset
31	I31	Interrupt associated with TCC #31	R	0x0
30	I30	Interrupt associated with TCC #30	R	0x0
29	I29	Interrupt associated with TCC #29	R	0x0
28	I28	Interrupt associated with TCC #28	R	0x0
27	I27	Interrupt associated with TCC #27	R	0x0
26	I26	Interrupt associated with TCC #26	R	0x0
25	I25	Interrupt associated with TCC #25	R	0x0
24	I24	Interrupt associated with TCC #24	R	0x0
23	I23	Interrupt associated with TCC #23	R	0x0
22	I22	Interrupt associated with TCC #22	R	0x0
21	I21	Interrupt associated with TCC #21	R	0x0
20	I20	Interrupt associated with TCC #20	R	0x0
19	I19	Interrupt associated with TCC #19	R	0x0
18	I18	Interrupt associated with TCC #18	R	0x0
17	I17	Interrupt associated with TCC #17	R	0x0
16	I16	Interrupt associated with TCC #16	R	0x0
15	I15	Interrupt associated with TCC #15	R	0x0
14	I14	Interrupt associated with TCC #14	R	0x0
13	I13	Interrupt associated with TCC #13	R	0x0
12	I12	Interrupt associated with TCC #12	R	0x0
11	I11	Interrupt associated with TCC #11	R	0x0
10	I10	Interrupt associated with TCC #10	R	0x0
9	I9	Interrupt associated with TCC #9	R	0x0
8	I8	Interrupt associated with TCC #8	R	0x0
7	I7	Interrupt associated with TCC #7	R	0x0
6	I6	Interrupt associated with TCC #6	R	0x0
5	I5	Interrupt associated with TCC #5	R	0x0
4	I4	Interrupt associated with TCC #4	R	0x0
3	I3	Interrupt associated with TCC #3	R	0x0
2	I2	Interrupt associated with TCC #2	R	0x0
1	I1	Interrupt associated with TCC #1	R	0x0
0	I0	Interrupt associated with TCC #0	R	0x0

Table 16-244. Register Call Summary for Register EDMA_TPCC_IPR

Enhanced DMA
- Dummy Versus Null Transfer Comparison: [0]
- Completion of a DMA Transfer: [1]
- Dummy or Null Completion: [2]
- Region Overview: [3]
- Transfer Completion Interrupts: [4] [5] [6] [7] [8] [9] [10] [11] [12] [13] [14] [15] [16] [17] [18] [19] [20] [21] [22] [23] [24] [25] [26] [27] [28] [29] [30] [31] [32] [33] [34] [35]
- EDMA Interrupt Servicing: [36] [37] [38] [39] [40]
- Interrupt Servicing: [41] [42] [43] [44] [45] [46]
- Interrupt Servicing: [47] [48] [49] [50] [51]
- Interrupt Evaluation Operations: [52]
- Event Dataflow: [53] [54]
- Ping-Pong Buffering: [55] [56] [57]
- Transfer Chaining Examples: [58]
- Setting Up an EDMA Transfer: [59] [60] [61] [62]
- EDMA Debug Checklist: [63] [64]
- EDMA Register Summary: [65] [66] [67] [68] [69] [70]
- EDMA Register Description: [71] [72] [73] [74] [75] [76] [77] [78] [79] [80] [81] [82] [83] [84] [85]

Table 16-245. EDMA_TPCC_IPRH

Address Offset	0x0000 106C		
Physical Address	0x4330 106C 0x40D1 106C 0x4151 106C 0x01D1 106C 0x420A 106C 0x421A 106C 0x422A 106C 0x423A 106C	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Interrupt Pending Register (High Part) EDMA_TPCC_IPRH.In bit is set when a interrupt completion code with TCC of N is detected. EDMA_TPCC_IPRH. In bit is cleared via software by writing a '1' to EDMA_TPCC_ICRH.In bit.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
I63	I62	I61	I60	I59	I58	I57	I56	I55	I54	I53	I52	I51	I50	I49	I48	I47	I46	I45	I44	I43	I42	I41	I40	I39	I38	I37	I36	I35	I34	I33	I32

Bits	Field Name	Description	Type	Reset
31	I63	Interrupt associated with TCC #63	R	0x0
30	I62	Interrupt associated with TCC #62	R	0x0
29	I61	Interrupt associated with TCC #61	R	0x0
28	I60	Interrupt associated with TCC #60	R	0x0
27	I59	Interrupt associated with TCC #59	R	0x0
26	I58	Interrupt associated with TCC #58	R	0x0
25	I57	Interrupt associated with TCC #57	R	0x0
24	I56	Interrupt associated with TCC #56	R	0x0
23	I55	Interrupt associated with TCC #55	R	0x0
22	I54	Interrupt associated with TCC #54	R	0x0
21	I53	Interrupt associated with TCC #53	R	0x0
20	I52	Interrupt associated with TCC #52	R	0x0
19	I51	Interrupt associated with TCC #51	R	0x0
18	I50	Interrupt associated with TCC #50	R	0x0

Bits	Field Name	Description	Type	Reset
17	I49	Interrupt associated with TCC #49	R	0x0
16	I48	Interrupt associated with TCC #48	R	0x0
15	I47	Interrupt associated with TCC #47	R	0x0
14	I46	Interrupt associated with TCC #46	R	0x0
13	I45	Interrupt associated with TCC #45	R	0x0
12	I44	Interrupt associated with TCC #44	R	0x0
11	I43	Interrupt associated with TCC #43	R	0x0
10	I42	Interrupt associated with TCC #42	R	0x0
9	I41	Interrupt associated with TCC #41	R	0x0
8	I40	Interrupt associated with TCC #40	R	0x0
7	I39	Interrupt associated with TCC #39	R	0x0
6	I38	Interrupt associated with TCC #38	R	0x0
5	I37	Interrupt associated with TCC #37	R	0x0
4	I36	Interrupt associated with TCC #36	R	0x0
3	I35	Interrupt associated with TCC #35	R	0x0
2	I34	Interrupt associated with TCC #34	R	0x0
1	I33	Interrupt associated with TCC #33	R	0x0
0	I32	Interrupt associated with TCC #32	R	0x0

Table 16-246. Register Call Summary for Register EDMA_TPCC_IPRH

Enhanced DMA

- Dummy Versus Null Transfer Comparison: [0]
- Dummy or Null Completion: [1]
- Region Overview: [2]
- Transfer Completion Interrupts: [3] [4] [5] [6] [7] [8] [9] [10] [11] [12] [13] [14] [15] [16] [17] [18] [19] [20] [21] [22] [23] [24]
- EDMA Interrupt Servicing: [25] [26] [27] [28] [29]
- Interrupt Servicing: [30] [31] [32] [33] [34] [35]
- Interrupt Servicing: [36] [37] [38] [39] [40]
- Interrupt Evaluation Operations: [41]
- Event Dataflow: [42] [43]
- Setting Up an EDMA Transfer: [44] [45] [46] [47]
- EDMA Debug Checklist: [48] [49]
- EDMA Register Summary: [50] [51] [52] [53] [54] [55]
- EDMA Register Description: [56] [57] [58] [59] [60] [61] [62] [63] [64] [65] [66] [67]

Table 16-247. EDMA_TPCC_ICR

Address Offset	0x0000 1070		
Physical Address	0x4330 1070 0x40D1 1070 0x4151 1070 0x01D1 1070 0x420A 1070 0x421A 1070 0x422A 1070 0x423A 1070	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Interrupt Clear Register CPU write of '1' to the EDMA_TPCC_ICR.In bit causes the EDMA_TPCC_IPR.In bit to be cleared. CPU write of '0' has no effect. All EDMA_TPCC_IPR.In bits must be cleared before additional interrupts will be asserted by CC.		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
I31	I30	I29	I28	I27	I26	I25	I24	I23	I22	I21	I20	I19	I18	I17	I16	I15	I14	I13	I12	I11	I10	I9	I8	I7	I6	I5	I4	I3	I2	I1	I0

Bits	Field Name	Description	Type	Reset
31	I31	Interrupt associated with TCC #31	W	0x0
30	I30	Interrupt associated with TCC #30	W	0x0
29	I29	Interrupt associated with TCC #29	W	0x0
28	I28	Interrupt associated with TCC #28	W	0x0
27	I27	Interrupt associated with TCC #27	W	0x0
26	I26	Interrupt associated with TCC #26	W	0x0
25	I25	Interrupt associated with TCC #25	W	0x0
24	I24	Interrupt associated with TCC #24	W	0x0
23	I23	Interrupt associated with TCC #23	W	0x0
22	I22	Interrupt associated with TCC #22	W	0x0
21	I21	Interrupt associated with TCC #21	W	0x0
20	I20	Interrupt associated with TCC #20	W	0x0
19	I19	Interrupt associated with TCC #19	W	0x0
18	I18	Interrupt associated with TCC #18	W	0x0
17	I17	Interrupt associated with TCC #17	W	0x0
16	I16	Interrupt associated with TCC #16	W	0x0
15	I15	Interrupt associated with TCC #15	W	0x0
14	I14	Interrupt associated with TCC #14	W	0x0
13	I13	Interrupt associated with TCC #13	W	0x0
12	I12	Interrupt associated with TCC #12	W	0x0
11	I11	Interrupt associated with TCC #11	W	0x0
10	I10	Interrupt associated with TCC #10	W	0x0
9	I9	Interrupt associated with TCC #9	W	0x0
8	I8	Interrupt associated with TCC #8	W	0x0
7	I7	Interrupt associated with TCC #7	W	0x0
6	I6	Interrupt associated with TCC #6	W	0x0
5	I5	Interrupt associated with TCC #5	W	0x0
4	I4	Interrupt associated with TCC #4	W	0x0
3	I3	Interrupt associated with TCC #3	W	0x0
2	I2	Interrupt associated with TCC #2	W	0x0
1	I1	Interrupt associated with TCC #1	W	0x0
0	I0	Interrupt associated with TCC #0	W	0x0

Table 16-248. Register Call Summary for Register EDMA_TPCC_ICR

Enhanced DMA

- Region Overview: [0]
- Transfer Completion Interrupts: [1] [2]
- Interrupt Servicing: [3]
- Setting Up an EDMA Transfer: [4] [5]
- EDMA Register Summary: [6] [7] [8] [9] [10] [11]
- EDMA Register Description: [12] [13] [14] [15]

Table 16-249. EDMA_TPCC_ICRH

Address Offset	0x0000 1074		
Physical Address	0x4330 1074 0x40D1 1074 0x4151 1074 0x01D1 1074 0x420A 1074 0x421A 1074 0x422A 1074 0x423A 1074	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Interrupt Clear Register (High Part) CPU write of '1' to the EDMA_TPCC_ICRH.In bit causes the EDMA_TPCC_IPRH.In bit to be cleared. CPU write of '0' has no effect. All EDMA_TPCC_IPRH.In bits must be cleared before additional interrupts will be asserted by CC.		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
I63	I62	I61	I60	I59	I58	I57	I56	I55	I54	I53	I52	I51	I50	I49	I48	I47	I46	I45	I44	I43	I42	I41	I40	I39	I38	I37	I36	I35	I34	I33	I32

Bits	Field Name	Description	Type	Reset
31	I63	Interrupt associated with TCC #63	W	0x0
30	I62	Interrupt associated with TCC #62	W	0x0
29	I61	Interrupt associated with TCC #61	W	0x0
28	I60	Interrupt associated with TCC #60	W	0x0
27	I59	Interrupt associated with TCC #59	W	0x0
26	I58	Interrupt associated with TCC #58	W	0x0
25	I57	Interrupt associated with TCC #57	W	0x0
24	I56	Interrupt associated with TCC #56	W	0x0
23	I55	Interrupt associated with TCC #55	W	0x0
22	I54	Interrupt associated with TCC #54	W	0x0
21	I53	Interrupt associated with TCC #53	W	0x0
20	I52	Interrupt associated with TCC #52	W	0x0
19	I51	Interrupt associated with TCC #51	W	0x0
18	I50	Interrupt associated with TCC #50	W	0x0
17	I49	Interrupt associated with TCC #49	W	0x0
16	I48	Interrupt associated with TCC #48	W	0x0
15	I47	Interrupt associated with TCC #47	W	0x0
14	I46	Interrupt associated with TCC #46	W	0x0
13	I45	Interrupt associated with TCC #45	W	0x0
12	I44	Interrupt associated with TCC #44	W	0x0
11	I43	Interrupt associated with TCC #43	W	0x0
10	I42	Interrupt associated with TCC #42	W	0x0
9	I41	Interrupt associated with TCC #41	W	0x0
8	I40	Interrupt associated with TCC #40	W	0x0
7	I39	Interrupt associated with TCC #39	W	0x0
6	I38	Interrupt associated with TCC #38	W	0x0
5	I37	Interrupt associated with TCC #37	W	0x0
4	I36	Interrupt associated with TCC #36	W	0x0
3	I35	Interrupt associated with TCC #35	W	0x0
2	I34	Interrupt associated with TCC #34	W	0x0
1	I33	Interrupt associated with TCC #33	W	0x0

Bits	Field Name	Description	Type	Reset
0	I32	Interrupt associated with TCC #32	W	0x0

Table 16-250. Register Call Summary for Register EDMA_TPCC_ICRH

Enhanced DMA

- Region Overview: [0]
- Transfer Completion Interrupts: [1]
- Interrupt Servicing: [2]
- Setting Up an EDMA Transfer: [3] [4]
- EDMA Register Summary: [5] [6] [7] [8] [9] [10]
- EDMA Register Description: [11] [12] [13] [14]

Table 16-251. EDMA_TPCC_IEVAL

Address Offset	0x0000 1078		
Physical Address	0x4330 1078 0x40D1 1078 0x4151 1078 0x01D1 1078 0x420A 1078 0x421A 1078 0x422A 1078 0x423A 1078	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Interrupt Eval Register		
Type	W		

<table>
<tr><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td colspan="30">RESERVED</td><td>SET</td><td>EVAL</td></tr>
</table>

Bits	Field Name	Description	Type	Reset
31:2	RESERVED	Reserved	R	0x0
1	SET	Interrupt Set: CPU write of '1' to the SETn bit causes the tpcc_intN output signal to be pulsed egardless of state of interrupts enable (IERn) and status (EDMA_TPCC_IPRn). CPU write of '0' has no effect.	W	0x0
0	EVAL	Interrupt Evaluate: CPU write of '1' to the EVALn bit causes the tpcc_intN output signal to be pulsed if any enabled interrupts (IERn) are still pending (EDMA_TPCC_IPRn). CPU write of '0' has no effect.	W	0x0

Table 16-252. Register Call Summary for Register EDMA_TPCC_IEVAL

Enhanced DMA

- Region Overview: [0] [1]
- Interrupt Servicing: [2] [3]
- Interrupt Evaluation Operations: [4] [5] [6] [7] [8]
- Error Interrupts: [9]
- EDMA Register Summary: [10] [11] [12] [13] [14] [15]

Table 16-253. EDMA_TPCC_QER

Address Offset	0x0000 1080		
Physical Address	0x4330 1080 0x40D1 1080 0x4151 1080 0x01D1 1080 0x420A 1080 0x421A 1080 0x422A 1080 0x423A 1080	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	QDMA Event Register: If EDMA_TPCC_QER.En bit is set, then the corresponding QDMA channel is prioritized vs. other qdma events for submission to the TC. EDMA_TPCC_QER.En bit is set when a vbus write byte matches the address defined in the QCHMAPn register. EDMA_TPCC_QER.En bit is cleared when the corresponding event is prioritized and serviced. EDMA_TPCC_QER.En is also cleared when user writes a '1' to the EDMA_TPCC_QSECR.En bit. If the EDMA_TPCC_QER.En bit is already set and a new QDMA event is detected due to user write to QDMA trigger location and EDMA_TPCC_QEER register is set, then the corresponding bit in the QDMA Event Missed Register is set.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	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

Bits	Field Name	Description	Type	Reset
31:8	RESERVED	Reserved	R	0x0
7	E7	Event #7	R	0x0
6	E6	Event #6	R	0x0
5	E5	Event #5	R	0x0
4	E4	Event #4	R	0x0
3	E3	Event #3	R	0x0
2	E2	Event #2	R	0x0
1	E1	Event #1	R	0x0
0	E0	Event #0	R	0x0

Table 16-254. Register Call Summary for Register EDMA_TPCC_QER

Enhanced DMA

- Dummy Versus Null Transfer Comparison: [0]
- Linking Transfers: [1]
- QDMA Channels: [2] [3] [4] [5]
- Region Overview: [6]
- Event Dataflow: [7] [8]
- Channel Priority: [9]
- EDMA Register Summary: [10] [11] [12] [13] [14] [15]
- EDMA Register Description: [16] [17] [18] [19] [20] [21] [22] [23] [24] [25] [26] [27] [28] [29] [30] [31] [32] [33] [34] [35] [36] [37]

Table 16-255. EDMA_TPCC_QEER

Address Offset	0x0000 1084		
Physical Address	0x4330 1084 0x40D1 1084 0x4151 1084 0x01D1 1084 0x420A 1084 0x421A 1084 0x422A 1084 0x423A 1084	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC

Table 16-255. EDMA_TPCC_QEER (continued)

Description	QDMA Event Enable Register Enabled/disabled QDMA address comparator for QDMA Channel N. EDMA_TPCC_QEER.En is not directly writeable. QDMA channels can be enabled via writes to EDMA_TPCC_QEESR and can be disabled via writes to EDMA_TPCC_QEECR register. EDMA_TPCC_QEER.En = 1, The corresponding QDMA channel comparator is enabled and Events will be recognized and latched in EDMA_TPCC_QER.En. EDMA_TPCC_QEER.En = 0, The corresponding QDMA channel comparator is disabled. Events will not be recognized/latched in EDMA_TPCC_QER.En.
Type	R

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	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

Bits	Field Name	Description	Type	Reset
31:8	RESERVED	Reserved	R Return 0's	0x0
7	E7	Event #7	R	0x0
6	E6	Event #6	R	0x0
5	E5	Event #5	R	0x0
4	E4	Event #4	R	0x0
3	E3	Event #3	R	0x0
2	E2	Event #2	R	0x0
1	E1	Event #1	R	0x0
0	E0	Event #0	R	0x0

Table 16-256. Register Call Summary for Register EDMA_TPCC_QEER

Enhanced DMA

- QDMA Channels: [0] [1]
- Region Overview: [2]
- Channel Controller Regions: [3]
- Setting Up an EDMA Transfer: [4]
- EDMA Debug Checklist: [5]
- EDMA Register Summary: [6] [7] [8] [9] [10] [11]
- EDMA Register Description: [12] [13] [14] [15] [16] [17] [18] [19] [20] [21]

Table 16-257. EDMA_TPCC_QEECR

Address Offset	0x0000 1088		
Physical Address	0x4330 1088 0x40D1 1088 0x4151 1088 0x01D1 1088 0x420A 1088 0x421A 1088 0x422A 1088 0x423A 1088	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	QDMA Event Enable Clear Register CPU write of '1' to the EDMA_TPCC_QEECR.En bit causes the EDMA_TPCC_QEER.En bit to be cleared. CPU write of '0' has no effect..		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	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

Bits	Field Name	Description	Type	Reset
31:8	RESERVED	Reserved	R	0x0
7	E7	Event #7	W	0x0
6	E6	Event #6	W	0x0
5	E5	Event #5	W	0x0
4	E4	Event #4	W	0x0
3	E3	Event #3	W	0x0
2	E2	Event #2	W	0x0
1	E1	Event #1	W	0x0
0	E0	Event #0	W	0x0

Table 16-258. Register Call Summary for Register EDMA_TPCC_QEECR

Enhanced DMA
- Region Overview: [0]
- EDMA Register Summary: [1] [2] [3] [4] [5] [6]
- EDMA Register Description: [7] [8] [9] [10]

Table 16-259. EDMA_TPCC_QEESR

Address Offset	0x0000 108C		
Physical Address	0x4330 108C 0x40D1 108C 0x4151 108C 0x01D1 108C 0x420A 108C 0x421A 108C 0x422A 108C 0x423A 108C	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	QDMA Event Enable Set Register CPU write of '1' to the EDMA_TPCC_QEESR.En bit causes the EDMA_TPCC_QEESR.En bit to be set. CPU write of '0' has no effect..		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	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

Bits	Field Name	Description	Type	Reset
31:8	RESERVED	Reserved	R	0x0
7	E7	Event #7	W	0x0
6	E6	Event #6	W	0x0
5	E5	Event #5	W	0x0
4	E4	Event #4	W	0x0
3	E3	Event #3	W	0x0
2	E2	Event #2	W	0x0
1	E1	Event #1	W	0x0
0	E0	Event #0	W	0x0

Table 16-260. Register Call Summary for Register EDMA_TPCC_QEESR

Enhanced DMA
- Region Overview: [0]
- Channel Controller Regions: [1]
- Setting Up an EDMA Transfer: [2]
- EDMA Register Summary: [3] [4] [5] [6] [7] [8]
- EDMA Register Description: [9] [10] [11] [12] [13] [14]

Table 16-261. EDMA_TPCC_QSER

Address Offset	0x0000 1090		
Physical Address	0x4330 1090 0x40D1 1090 0x4151 1090 0x01D1 1090 0x420A 1090 0x421A 1090 0x422A 1090 0x423A 1090	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	QDMA Secondary Event Register The QDMA secondary event register is used along with the QDMA Event Register (EDMA_TPCC_QER) to provide information on the state of a QDMA Event. En = 0 : Event is not currently in the Event Queue. En = 1 : Event is currently stored in Event Queue. Event arbiter will not prioritize additional events.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	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

Bits	Field Name	Description	Type	Reset
31:8	RESERVED	Reserved	R Return 0's	0x0
7	E7	Event #7	R	0x0
6	E6	Event #6	R	0x0
5	E5	Event #5	R	0x0
4	E4	Event #4	R	0x0
3	E3	Event #3	R	0x0
2	E2	Event #2	R	0x0
1	E1	Event #1	R	0x0
0	E0	Event #0	R	0x0

Table 16-262. Register Call Summary for Register EDMA_TPCC_QSER

Enhanced DMA
- Null PaRAM Set: [0] [1]
- Dummy PaRAM Set: [2]
- Dummy Versus Null Transfer Comparison: [3]
- Event Dataflow: [4]
- EDMA Debug Checklist: [5] [6] [7]
- EDMA Register Summary: [8] [9] [10] [11] [12] [13]
- EDMA Register Description: [14] [15] [16] [17]

Table 16-263. EDMA_TPCC_QSECR

Address Offset	0x0000 1094		
Physical Address	0x4330 1094 0x40D1 1094 0x4151 1094 0x01D1 1094 0x420A 1094 0x421A 1094 0x422A 1094 0x423A 1094	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	QDMA Secondary Event Clear Register The secondary event clear register is used to clear the status of the EDMA_TPCC_QSER and EDMA_TPCC_QER register (note that this is slightly different than the EDMA_TPCC_SER operation, which does not clear the EDMA_TPCC_ER.En register). CPU write of '1' to the EDMA_TPCC_QSECR.En bit clears the EDMA_TPCC_QSER.En and EDMA_TPCC_QER.En register fields. CPU write of '0' has no effect..		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	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

Bits	Field Name	Description	Type	Reset
31:8	RESERVED	Reserved	R	0x0
7	E7	Event #7	W	0x0
6	E6	Event #6	W	0x0
5	E5	Event #5	W	0x0
4	E4	Event #4	W	0x0
3	E3	Event #3	W	0x0
2	E2	Event #2	W	0x0
1	E1	Event #1	W	0x0
0	E0	Event #0	W	0x0

Table 16-264. Register Call Summary for Register EDMA_TPCC_QSECR

Enhanced DMA
- EDMA Register Summary: [0] [1] [2] [3] [4] [5]
- EDMA Register Description: [6] [7] [8] [9]

Table 16-265. EDMA_TPCC_ER_RN_k

Address Offset	0x0000 2000 + (0x200 * k)		
Physical Address	0x4330 2000 + (0x200 * k) 0x40D1 2000 + (0x200 * k) 0x4151 2000 + (0x200 * k) 0x01D1 2000 + (0x200 * k) 0x420A 2000 + (0x200 * k) 0x421A 2000 + (0x200 * k) 0x422A 2000 + (0x200 * k) 0x423A 2000 + (0x200 * k)	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC

Table 16-265. EDMA_TPCC_ER_RN_k (continued)

Description	Event Register If EDMA_TPCC_ER.En bit is set and the EDMA_TPCC_EER.En bit is also set, then the corresponding DMA channel is prioritized vs. other pending DMA events for submission to the TC. EDMA_TPCC_ER.En bit is set when the input event #n transitions from inactive (low) to active (high), regardless of the state of EDMA_TPCC_EER.En bit. EDMA_TPCC_ER.En bit is cleared when the corresponding event is prioritized and serviced. If the EDMA_TPCC_ER.En bit is already set and a new inactive to active transition is detected on the input event #n input AND the corresponding bit in the EDMA_TPCC_EER register is set, then the corresponding bit in the Event Missed Register is set. Event N can be cleared via sw by writing a '1' to the EDMA_TPCC_ECR pseudo-register.
Type	R

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E31	E30	E29	E28	E27	E26	E25	E24	E23	E22	E21	E20	E19	E18	E17	E16	E15	E14	E13	E12	E11	E10	E9	E8	E7	E6	E5	E4	E3	E2	E1	E0

Bits	Field Name	Description	Type	Reset
31	E31	Event #31	R	0x0
30	E30	Event #30	R	0x0
29	E29	Event #29	R	0x0
28	E28	Event #28	R	0x0
27	E27	Event #27	R	0x0
26	E26	Event #26	R	0x0
25	E25	Event #25	R	0x0
24	E24	Event #24	R	0x0
23	E23	Event #23	R	0x0
22	E22	Event #22	R	0x0
21	E21	Event #21	R	0x0
20	E20	Event #20	R	0x0
19	E19	Event #19	R	0x0
18	E18	Event #18	R	0x0
17	E17	Event #17	R	0x0
16	E16	Event #16	R	0x0
15	E15	Event #15	R	0x0
14	E14	Event #14	R	0x0
13	E13	Event #13	R	0x0
12	E12	Event #12	R	0x0
11	E11	Event #11	R	0x0
10	E10	Event #10	R	0x0
9	E9	Event #9	R	0x0
8	E8	Event #8	R	0x0
7	E7	Event #7	R	0x0
6	E6	Event #6	R	0x0
5	E5	Event #5	R	0x0
4	E4	Event #4	R	0x0
3	E3	Event #3	R	0x0
2	E2	Event #2	R	0x0
1	E1	Event #1	R	0x0
0	E0	Event #0	R	0x0

Table 16-266. Register Call Summary for Register EDMA_TPCC_ER_RN_k

Enhanced DMA
- EDMA Register Summary: [0] [1] [2] [3] [4] [5]

Table 16-267. EDMA_TPCC_ERH_RN_k

Address Offset	0x0000 2004 + (0x200 * k)		
Physical Address	0x4330 2004 + (0x200 * k) 0x40D1 2004 + (0x200 * k) 0x4151 2004 + (0x200 * k) 0x01D1 2004 + (0x200 * k) 0x420A 2004 + (0x200 * k) 0x421A 2004 + (0x200 * k) 0x422A 2004 + (0x200 * k) 0x423A 2004 + (0x200 * k)	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Event Register (High Part) If EDMA_TPCC_ERH.En bit is set and the EDMA_TPCC_EERH.En bit is also set, then the corresponding DMA channel is prioritized vs. other pending DMA events for submission to the TC. EDMA_TPCC_ERH.En bit is set when the input event #n transitions from inactive (low) to active (high), regardless of the state of EERH.En bit. EDMA_TPCC_ER.En bit is cleared when the corresponding event is prioritized and serviced. If the EDMA_TPCC_ERH.En bit is already set and a new inactive to active transition is detected on the input event #n input AND the corresponding bit in the EDMA_TPCC_EERH register is set, then the corresponding bit in the Event Missed Register is set. Event N can be cleared via sw by writing a '1' to the EDMA_TPCC_ECRH pseudo-register.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E63	E62	E61	E60	E59	E58	E57	E56	E55	E54	E53	E52	E51	E50	E49	E48	E47	E46	E45	E44	E43	E42	E41	E40	E39	E38	E37	E36	E35	E34	E33	E32

Bits	Field Name	Description	Type	Reset
31	E63	Event #63	R	0x0
30	E62	Event #62	R	0x0
29	E61	Event #61	R	0x0
28	E60	Event #60	R	0x0
27	E59	Event #59	R	0x0
26	E58	Event #58	R	0x0
25	E57	Event #57	R	0x0
24	E56	Event #56	R	0x0
23	E55	Event #55	R	0x0
22	E54	Event #54	R	0x0
21	E53	Event #53	R	0x0
20	E52	Event #52	R	0x0
19	E51	Event #51	R	0x0
18	E50	Event #50	R	0x0
17	E49	Event #49	R	0x0
16	E48	Event #48	R	0x0
15	E47	Event #47	R	0x0
14	E46	Event #46	R	0x0
13	E45	Event #45	R	0x0
12	E44	Event #44	R	0x0
11	E43	Event #43	R	0x0
10	E42	Event #42	R	0x0
9	E41	Event #41	R	0x0

Bits	Field Name	Description	Type	Reset
8	E40	Event #40	R	0x0
7	E39	Event #39	R	0x0
6	E38	Event #38	R	0x0
5	E37	Event #37	R	0x0
4	E36	Event #36	R	0x0
3	E35	Event #35	R	0x0
2	E34	Event #34	R	0x0
1	E33	Event #33	R	0x0
0	E32	Event #32	R	0x0

Table 16-268. Register Call Summary for Register EDMA_TPCC_ERH_RN_k

Enhanced DMA

- EDMA Register Summary: [0] [1] [2] [3] [4] [5]

Table 16-269. EDMA_TPCC_ECR_RN_k

Address Offset	0x0000 2008 + (0x200 * k)		
Physical Address	0x4330 2008 + (0x200 * k) 0x40D1 2008 + (0x200 * k) 0x4151 2008 + (0x200 * k) 0x01D1 2008 + (0x200 * k) 0x420A 2008 + (0x200 * k) 0x421A 2008 + (0x200 * k) 0x422A 2008 + (0x200 * k) 0x423A 2008 + (0x200 * k)	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Event Clear Register CPU write of '1' to the EDMA_TPCC_ECR.En bit causes the EDMA_TPCC_ER.En bit to be cleared. CPU write of '0' has no effect.		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E31	E30	E29	E28	E27	E26	E25	E24	E23	E22	E21	E20	E19	E18	E17	E16	E15	E14	E13	E12	E11	E10	E9	E8	E7	E6	E5	E4	E3	E2	E1	E0

Bits	Field Name	Description	Type	Reset
31	E31	Event #31	W	0x0
30	E30	Event #30	W	0x0
29	E29	Event #29	W	0x0
28	E28	Event #28	W	0x0
27	E27	Event #27	W	0x0
26	E26	Event #26	W	0x0
25	E25	Event #25	W	0x0
24	E24	Event #24	W	0x0
23	E23	Event #23	W	0x0
22	E22	Event #22	W	0x0
21	E21	Event #21	W	0x0
20	E20	Event #20	W	0x0
19	E19	Event #19	W	0x0
18	E18	Event #18	W	0x0
17	E17	Event #17	W	0x0
16	E16	Event #16	W	0x0

Bits	Field Name	Description	Type	Reset
15	E15	Event #15	W	0x0
14	E14	Event #14	W	0x0
13	E13	Event #13	W	0x0
12	E12	Event #12	W	0x0
11	E11	Event #11	W	0x0
10	E10	Event #10	W	0x0
9	E9	Event #9	W	0x0
8	E8	Event #8	W	0x0
7	E7	Event #7	W	0x0
6	E6	Event #6	W	0x0
5	E5	Event #5	W	0x0
4	E4	Event #4	W	0x0
3	E3	Event #3	W	0x0
2	E2	Event #2	W	0x0
1	E1	Event #1	W	0x0
0	E0	Event #0	W	0x0

Table 16-270. Register Call Summary for Register EDMA_TPCC_ECR_RN_k

Enhanced DMA
- EDMA Register Summary: [0] [1] [2] [3] [4] [5]

Table 16-271. EDMA_TPCC_ECRH_RN_k

Address Offset	0x0000 200C + (0x200 * k)		
Physical Address	0x4330 200C + (0x200 * k) 0x40D1 200C + (0x200 * k) 0x4151 200C + (0x200 * k) 0x01D1 200C + (0x200 * k) 0x420A 200C + (0x200 * k) 0x421A 200C + (0x200 * k) 0x422A 200C + (0x200 * k) 0x423A 200C + (0x200 * k)	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Event Clear Register (High Part) CPU write of '1' to the EDMA_TPCC_ECRH.En bit causes the EDMA_TPCC_ERH.En bit to be cleared. CPU write of '0' has no effect.		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E63	E62	E61	E60	E59	E58	E57	E56	E55	E54	E53	E52	E51	E50	E49	E48	E47	E46	E45	E44	E43	E42	E41	E40	E39	E38	E37	E36	E35	E34	E33	E32

Bits	Field Name	Description	Type	Reset
31	E63	Event #63	W	0x0
30	E62	Event #62	W	0x0
29	E61	Event #61	W	0x0
28	E60	Event #60	W	0x0
27	E59	Event #59	W	0x0
26	E58	Event #58	W	0x0
25	E57	Event #57	W	0x0
24	E56	Event #56	W	0x0
23	E55	Event #55	W	0x0

Bits	Field Name	Description	Type	Reset
22	E54	Event #54	W	0x0
21	E53	Event #53	W	0x0
20	E52	Event #52	W	0x0
19	E51	Event #51	W	0x0
18	E50	Event #50	W	0x0
17	E49	Event #49	W	0x0
16	E48	Event #48	W	0x0
15	E47	Event #47	W	0x0
14	E46	Event #46	W	0x0
13	E45	Event #45	W	0x0
12	E44	Event #44	W	0x0
11	E43	Event #43	W	0x0
10	E42	Event #42	W	0x0
9	E41	Event #41	W	0x0
8	E40	Event #40	W	0x0
7	E39	Event #39	W	0x0
6	E38	Event #38	W	0x0
5	E37	Event #37	W	0x0
4	E36	Event #36	W	0x0
3	E35	Event #35	W	0x0
2	E34	Event #34	W	0x0
1	E33	Event #33	W	0x0
0	E32	Event #32	W	0x0

Table 16-272. Register Call Summary for Register EDMA_TPCC_ECRH_RN_k

Enhanced DMA

- EDMA Register Summary: [0] [1] [2] [3] [4] [5]

Table 16-273. EDMA_TPCC_ESR_RN_k

Address Offset	0x0000 2010 + (0x200 * k)		
Physical Address	0x4330 2010 + (0x200 * k) 0x40D1 2010 + (0x200 * k) 0x4151 2010 + (0x200 * k) 0x01D1 2010 + (0x200 * k) 0x420A 2010 + (0x200 * k) 0x421A 2010 + (0x200 * k) 0x422A 2010 + (0x200 * k) 0x423A 2010 + (0x200 * k)	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Event Set Register CPU write of '1' to the EDMA_TPCC_ESR.En bit causes the EDMA_TPCC_ER.En bit to be set. CPU write of '0' has no effect.		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E31	E30	E29	E28	E27	E26	E25	E24	E23	E22	E21	E20	E19	E18	E17	E16	E15	E14	E13	E12	E11	E10	E9	E8	E7	E6	E5	E4	E3	E2	E1	E0

Bits	Field Name	Description	Type	Reset
31	E31	Event #31	W	0x0
30	E30	Event #30	W	0x0
29	E29	Event #29	W	0x0

Bits	Field Name	Description	Type	Reset
28	E28	Event #28	W	0x0
27	E27	Event #27	W	0x0
26	E26	Event #26	W	0x0
25	E25	Event #25	W	0x0
24	E24	Event #24	W	0x0
23	E23	Event #23	W	0x0
22	E22	Event #22	W	0x0
21	E21	Event #21	W	0x0
20	E20	Event #20	W	0x0
19	E19	Event #19	W	0x0
18	E18	Event #18	W	0x0
17	E17	Event #17	W	0x0
16	E16	Event #16	W	0x0
15	E15	Event #15	W	0x0
14	E14	Event #14	W	0x0
13	E13	Event #13	W	0x0
12	E12	Event #12	W	0x0
11	E11	Event #11	W	0x0
10	E10	Event #10	W	0x0
9	E9	Event #9	W	0x0
8	E8	Event #8	W	0x0
7	E7	Event #7	W	0x0
6	E6	Event #6	W	0x0
5	E5	Event #5	W	0x0
4	E4	Event #4	W	0x0
3	E3	Event #3	W	0x0
2	E2	Event #2	W	0x0
1	E1	Event #1	W	0x0
0	E0	Event #0	W	0x0

Table 16-274. Register Call Summary for Register EDMA_TPCC_ESR_RN_k

Enhanced DMA

- EDMA Register Summary: [0] [1] [2] [3] [4] [5]

Table 16-275. EDMA_TPCC_ESRH_RN_k

Address Offset	0x0000 2014 + (0x200 * k)		
Physical Address	0x4330 2014 + (0x200 * k) 0x40D1 2014 + (0x200 * k) 0x4151 2014 + (0x200 * k) 0x01D1 2014 + (0x200 * k) 0x420A 2014 + (0x200 * k) 0x421A 2014 + (0x200 * k) 0x422A 2014 + (0x200 * k) 0x423A 2014 + (0x200 * k)	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Event Set Register (High Part) CPU write of '1' to the EDMA_TPCC_ESRH.En bit causes the EDMA_TPCC_ERH.En bit to be set. CPU write of '0' has no effect.		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E63	E62	E61	E60	E59	E58	E57	E56	E55	E54	E53	E52	E51	E50	E49	E48	E47	E46	E45	E44	E43	E42	E41	E40	E39	E38	E37	E36	E35	E34	E33	E32

Bits	Field Name	Description	Type	Reset
31	E63	Event #63	W	0x0
30	E62	Event #62	W	0x0
29	E61	Event #61	W	0x0
28	E60	Event #60	W	0x0
27	E59	Event #59	W	0x0
26	E58	Event #58	W	0x0
25	E57	Event #57	W	0x0
24	E56	Event #56	W	0x0
23	E55	Event #55	W	0x0
22	E54	Event #54	W	0x0
21	E53	Event #53	W	0x0
20	E52	Event #52	W	0x0
19	E51	Event #51	W	0x0
18	E50	Event #50	W	0x0
17	E49	Event #49	W	0x0
16	E48	Event #48	W	0x0
15	E47	Event #47	W	0x0
14	E46	Event #46	W	0x0
13	E45	Event #45	W	0x0
12	E44	Event #44	W	0x0
11	E43	Event #43	W	0x0
10	E42	Event #42	W	0x0
9	E41	Event #41	W	0x0
8	E40	Event #40	W	0x0
7	E39	Event #39	W	0x0
6	E38	Event #38	W	0x0
5	E37	Event #37	W	0x0
4	E36	Event #36	W	0x0
3	E35	Event #35	W	0x0
2	E34	Event #34	W	0x0
1	E33	Event #33	W	0x0
0	E32	Event #32	W	0x0

Table 16-276. Register Call Summary for Register EDMA_TPCC_ESRH_RN_k

Enhanced DMA

- EDMA Register Summary: [0] [1] [2] [3] [4] [5]

Table 16-277. EDMA_TPCC_CER_RN_k

Address Offset	0x0000 2018 + (0x200 * k)		
Physical Address	0x4330 2018 + (0x200 * k) 0x40D1 2018 + (0x200 * k) 0x4151 2018 + (0x200 * k) 0x01D1 2018 + (0x200 * k) 0x420A 2018 + (0x200 * k) 0x421A 2018 + (0x200 * k) 0x422A 2018 + (0x200 * k) 0x423A 2018 + (0x200 * k)	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC

Table 16-277. EDMA_TPCC_CER_RN_k (continued)

Description	Chained Event Register If EDMA_TPCC_CER.En bit is set (regardless of state of EDMA_TPCC_EER.En), then the corresponding DMA channel is prioritized vs. other pending DMA events for submission to the TC. EDMA_TPCC_CER.En bit is set when a chaining completion code is returned from one of the 3PTCs via the completion interface, or is generated internally via Early Completion path. EDMA_TPCC_CER.En bit is cleared when the corresponding event is prioritized and serviced. If the EDMA_TPCC_CER.En bit is already set and the corresponding chaining completion code is returned from the TC, then the corresponding bit in the Event Missed Register is set. EDMA_TPCC_CER.En cannot be set or cleared via software.
Type	R

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E31	E30	E29	E28	E27	E26	E25	E24	E23	E22	E21	E20	E19	E18	E17	E16	E15	E14	E13	E12	E11	E10	E9	E8	E7	E6	E5	E4	E3	E2	E1	E0

Bits	Field Name	Description	Type	Reset
31	E31	Event #31	R	0x0
30	E30	Event #30	R	0x0
29	E29	Event #29	R	0x0
28	E28	Event #28	R	0x0
27	E27	Event #27	R	0x0
26	E26	Event #26	R	0x0
25	E25	Event #25	R	0x0
24	E24	Event #24	R	0x0
23	E23	Event #23	R	0x0
22	E22	Event #22	R	0x0
21	E21	Event #21	R	0x0
20	E20	Event #20	R	0x0
19	E19	Event #19	R	0x0
18	E18	Event #18	R	0x0
17	E17	Event #17	R	0x0
16	E16	Event #16	R	0x0
15	E15	Event #15	R	0x0
14	E14	Event #14	R	0x0
13	E13	Event #13	R	0x0
12	E12	Event #12	R	0x0
11	E11	Event #11	R	0x0
10	E10	Event #10	R	0x0
9	E9	Event #9	R	0x0
8	E8	Event #8	R	0x0
7	E7	Event #7	R	0x0
6	E6	Event #6	R	0x0
5	E5	Event #5	R	0x0
4	E4	Event #4	R	0x0
3	E3	Event #3	R	0x0
2	E2	Event #2	R	0x0
1	E1	Event #1	R	0x0
0	E0	Event #0	R	0x0

Table 16-278. Register Call Summary for Register EDMA_TPCC_CER_RN_k

Enhanced DMA

- EDMA Register Summary: [0] [1] [2] [3] [4] [5]

Table 16-279. EDMA_TPCC_CERH_RN_k

Address Offset	0x0000 201C + (0x200 * k)		
Physical Address	0x4330 201C + (0x200 * k) 0x40D1 201C + (0x200 * k) 0x4151 201C + (0x200 * k) 0x01D1 201C + (0x200 * k) 0x420A 201C + (0x200 * k) 0x421A 201C + (0x200 * k) 0x422A 201C + (0x200 * k) 0x423A 201C + (0x200 * k)	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Chained Event Register (High Part) If EDMA_TPCC_CERH.En bit is set (regardless of state of EDMA_TPCC_EERH.En), then the corresponding DMA channel is prioritized vs. other pending DMA events for submission to the TC. EDMA_TPCC_CERH.En bit is set when a chaining completion code is returned from one of the 3PTCs via the completion interface, or is generated internally via Early Completion path. EDMA_TPCC_CERH.En bit is cleared when the corresponding event is prioritized and serviced. If the EDMA_TPCC_CERH.En bit is already set and the corresponding chaining completion code is returned from the TC, then the corresponding bit in the Event Missed Register is set. EDMA_TPCC_CERH.En cannot be set or cleared via software.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E63	E62	E61	E60	E59	E58	E57	E56	E55	E54	E53	E52	E51	E50	E49	E48	E47	E46	E45	E44	E43	E42	E41	E40	E39	E38	E37	E36	E35	E34	E33	E32

Bits	Field Name	Description	Type	Reset
31	E63	Event #63	R	0x0
30	E62	Event #62	R	0x0
29	E61	Event #61	R	0x0
28	E60	Event #60	R	0x0
27	E59	Event #59	R	0x0
26	E58	Event #58	R	0x0
25	E57	Event #57	R	0x0
24	E56	Event #56	R	0x0
23	E55	Event #55	R	0x0
22	E54	Event #54	R	0x0
21	E53	Event #53	R	0x0
20	E52	Event #52	R	0x0
19	E51	Event #51	R	0x0
18	E50	Event #50	R	0x0
17	E49	Event #49	R	0x0
16	E48	Event #48	R	0x0
15	E47	Event #47	R	0x0
14	E46	Event #46	R	0x0
13	E45	Event #45	R	0x0
12	E44	Event #44	R	0x0
11	E43	Event #43	R	0x0
10	E42	Event #42	R	0x0
9	E41	Event #41	R	0x0
8	E40	Event #40	R	0x0

Bits	Field Name	Description	Type	Reset
7	E39	Event #39	R	0x0
6	E38	Event #38	R	0x0
5	E37	Event #37	R	0x0
4	E36	Event #36	R	0x0
3	E35	Event #35	R	0x0
2	E34	Event #34	R	0x0
1	E33	Event #33	R	0x0
0	E32	Event #32	R	0x0

Table 16-280. Register Call Summary for Register EDMA_TPCC_CERH_RN_k

Enhanced DMA
- EDMA Register Summary: [0] [1] [2] [3] [4] [5]

Table 16-281. EDMA_TPCC_EER_RN_k

Address Offset	0x0000 2020 + (0x200 * k)		
Physical Address	0x4330 2020 + (0x200 * k) 0x40D1 2020 + (0x200 * k) 0x4151 2020 + (0x200 * k) 0x01D1 2020 + (0x200 * k) 0x420A 2020 + (0x200 * k) 0x421A 2020 + (0x200 * k) 0x422A 2020 + (0x200 * k) 0x423A 2020 + (0x200 * k)	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Event Enable Register Enables DMA transfers for EDMA_TPCC_ER.En pending events. EDMA_TPCC_ER.En is set based on externally asserted events (via tpcc_eventN_pi). This register has no effect on Chained Event Register (EDMA_TPCC_CER) or Event Set Register (EDMA_TPCC_ESR). **NOTE: If a bit is set in EDMA_TPCC_ER.En while EDMA_TPCC_EER.En is disabled, no action is taken.** If EDMA_TPCC_EER.En is enabled at a later point (and EDMA_TPCC_ER.En has not been cleared via SW) then the event will be recognized as a valid 'TR Sync' EDMA_TPCC_EER.En is not directly writeable. Events can be enabled via writes to EDMA_TPCC_EESR and can be disabled via writes to EDMA_TPCC_EECR register. EDMA_TPCC_EER.En = 0: EDMA_TPCC_ER.En is not enabled to trigger DMA transfers. EDMA_TPCC_EER.En = 1: EDMA_TPCC_ER.En is enabled to trigger DMA transfers.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E31	E30	E29	E28	E27	E26	E25	E24	E23	E22	E21	E20	E19	E18	E17	E16	E15	E14	E13	E12	E11	E10	E9	E8	E7	E6	E5	E4	E3	E2	E1	E0

Bits	Field Name	Description	Type	Reset
31	E31	Event #31	R	0x0
30	E30	Event #30	R	0x0
29	E29	Event #29	R	0x0
28	E28	Event #28	R	0x0
27	E27	Event #27	R	0x0
26	E26	Event #26	R	0x0
25	E25	Event #25	R	0x0
24	E24	Event #24	R	0x0
23	E23	Event #23	R	0x0
22	E22	Event #22	R	0x0
21	E21	Event #21	R	0x0
20	E20	Event #20	R	0x0

Bits	Field Name	Description	Type	Reset
19	E19	Event #19	R	0x0
18	E18	Event #18	R	0x0
17	E17	Event #17	R	0x0
16	E16	Event #16	R	0x0
15	E15	Event #15	R	0x0
14	E14	Event #14	R	0x0
13	E13	Event #13	R	0x0
12	E12	Event #12	R	0x0
11	E11	Event #11	R	0x0
10	E10	Event #10	R	0x0
9	E9	Event #9	R	0x0
8	E8	Event #8	R	0x0
7	E7	Event #7	R	0x0
6	E6	Event #6	R	0x0
5	E5	Event #5	R	0x0
4	E4	Event #4	R	0x0
3	E3	Event #3	R	0x0
2	E2	Event #2	R	0x0
1	E1	Event #1	R	0x0
0	E0	Event #0	R	0x0

Table 16-282. Register Call Summary for Register EDMA_TPCC_EER_RN_k

Enhanced DMA

- EDMA Register Summary: [0] [1] [2] [3] [4] [5]

Table 16-283. EDMA_TPCC_EERH_RN_k

Address Offset	0x0000 2024 + (0x200 * k)		
Physical Address	0x4330 2024 + (0x200 * k) 0x40D1 2024 + (0x200 * k) 0x4151 2024 + (0x200 * k) 0x01D1 2024 + (0x200 * k) 0x420A 2024 + (0x200 * k) 0x421A 2024 + (0x200 * k) 0x422A 2024 + (0x200 * k) 0x423A 2024 + (0x200 * k)	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Event Enable Register (High Part) Enables DMA transfers for EDMA_TPCC_ERH.En pending events. EDMA_TPCC_ERH.En is set based on externally asserted events (via tpcc_eventN_pi). This register has no effect on Chained Event Register (EDMA_TPCC_CERH) or Event Set Register (EDMA_TPCC_ESRH). **NOTE: If a bit is set in EDMA_TPCC_ERH.En while EDMA_TPCC_EERH.En is disabled, no action is taken.** If EDMA_TPCC_EERH.En is enabled at a later point (and EDMA_TPCC_ERH.En has not been cleared via SW) then the event will be recognized as a valid 'TR Sync' EDMA_TPCC_EERH.En is not directly writeable. Events can be enabled via writes to EDMA_TPCC_EESRH and can be disabled via writes to EDMA_TPCC_EECRH register. EDMA_TPCC_EERH.En = 0: EDMA_TPCC_ER.En is not enabled to trigger DMA transfers. EDMA_TPCC_EERH.En = 1: EDMA_TPCC_ER.En is enabled to trigger DMA transfers.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E63	E62	E61	E60	E59	E58	E57	E56	E55	E54	E53	E52	E51	E50	E49	E48	E47	E46	E45	E44	E43	E42	E41	E40	E39	E38	E37	E36	E35	E34	E33	E32

Bits	Field Name	Description	Type	Reset
31	E63	Event #63	R	0x0
30	E62	Event #62	R	0x0
29	E61	Event #61	R	0x0
28	E60	Event #60	R	0x0
27	E59	Event #59	R	0x0
26	E58	Event #58	R	0x0
25	E57	Event #57	R	0x0
24	E56	Event #56	R	0x0
23	E55	Event #55	R	0x0
22	E54	Event #54	R	0x0
21	E53	Event #53	R	0x0
20	E52	Event #52	R	0x0
19	E51	Event #51	R	0x0
18	E50	Event #50	R	0x0
17	E49	Event #49	R	0x0
16	E48	Event #48	R	0x0
15	E47	Event #47	R	0x0
14	E46	Event #46	R	0x0
13	E45	Event #45	R	0x0
12	E44	Event #44	R	0x0
11	E43	Event #43	R	0x0
10	E42	Event #42	R	0x0
9	E41	Event #41	R	0x0
8	E40	Event #40	R	0x0
7	E39	Event #39	R	0x0
6	E38	Event #38	R	0x0
5	E37	Event #37	R	0x0
4	E36	Event #36	R	0x0
3	E35	Event #35	R	0x0
2	E34	Event #34	R	0x0
1	E33	Event #33	R	0x0
0	E32	Event #32	R	0x0

Table 16-284. Register Call Summary for Register EDMA_TPCC_EERH_RN_k

Enhanced DMA

- EDMA Register Summary: [0] [1] [2] [3] [4] [5]

Table 16-285. EDMA_TPCC_EECR_RN_k

Address Offset	0x0000 2028 + (0x200 * k)		
Physical Address	0x4330 2028 + (0x200 * k) 0x40D1 2028 + (0x200 * k) 0x4151 2028 + (0x200 * k) 0x01D1 2028 + (0x200 * k) 0x420A 2028 + (0x200 * k) 0x421A 2028 + (0x200 * k) 0x422A 2028 + (0x200 * k) 0x423A 2028 + (0x200 * k)	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Event Enable Clear Register CPU write of '0' has no effect. CPU write of '1' to the EDMA_TPCC_EECR.En bit causes the EDMA_TPCC_EER.En bit to be cleared.		

Table 16-285. EDMA_TPCC_EECR_RN_k (continued)

Type	W

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E31	E30	E29	E28	E27	E26	E25	E24	E23	E22	E21	E20	E19	E18	E17	E16	E15	E14	E13	E12	E11	E10	E9	E8	E7	E6	E5	E4	E3	E2	E1	E0

Bits	Field Name	Description	Type	Reset
31	E31	Event #31	W	0x0
30	E30	Event #30	W	0x0
29	E29	Event #29	W	0x0
28	E28	Event #28	W	0x0
27	E27	Event #27	W	0x0
26	E26	Event #26	W	0x0
25	E25	Event #25	W	0x0
24	E24	Event #24	W	0x0
23	E23	Event #23	W	0x0
22	E22	Event #22	W	0x0
21	E21	Event #21	W	0x0
20	E20	Event #20	W	0x0
19	E19	Event #19	W	0x0
18	E18	Event #18	W	0x0
17	E17	Event #17	W	0x0
16	E16	Event #16	W	0x0
15	E15	Event #15	W	0x0
14	E14	Event #14	W	0x0
13	E13	Event #13	W	0x0
12	E12	Event #12	W	0x0
11	E11	Event #11	W	0x0
10	E10	Event #10	W	0x0
9	E9	Event #9	W	0x0
8	E8	Event #8	W	0x0
7	E7	Event #7	W	0x0
6	E6	Event #6	W	0x0
5	E5	Event #5	W	0x0
4	E4	Event #4	W	0x0
3	E3	Event #3	W	0x0
2	E2	Event #2	W	0x0
1	E1	Event #1	W	0x0
0	E0	Event #0	W	0x0

Table 16-286. Register Call Summary for Register EDMA_TPCC_EECR_RN_k

Enhanced DMA

- EDMA Register Summary: [0] [1] [2] [3] [4] [5]

Table 16-287. EDMA_TPCC_EECRH_RN_k

Address Offset	0x0000 202C + (0x200 * k)		
Physical Address	0x4330 202C + (0x200 * k) 0x40D1 202C + (0x200 * k) 0x4151 202C + (0x200 * k) 0x01D1 202C + (0x200 * k) 0x420A 202C + (0x200 * k) 0x421A 202C + (0x200 * k) 0x422A 202C + (0x200 * k) 0x423A 202C + (0x200 * k)	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Event Enable Clear Register (High Part) CPU write of '0' has no effect. CPU write of '1' to the EDMA_TPCC_EECRH.En bit causes the EDMA_TPCC_EERH.En bit to be cleared.		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E63	E62	E61	E60	E59	E58	E57	E56	E55	E54	E53	E52	E51	E50	E49	E48	E47	E46	E45	E44	E43	E42	E41	E40	E39	E38	E37	E36	E35	E34	E33	E32

Bits	Field Name	Description	Type	Reset
31	E63	Event #63	W	0x0
30	E62	Event #62	W	0x0
29	E61	Event #61	W	0x0
28	E60	Event #60	W	0x0
27	E59	Event #59	W	0x0
26	E58	Event #58	W	0x0
25	E57	Event #57	W	0x0
24	E56	Event #56	W	0x0
23	E55	Event #55	W	0x0
22	E54	Event #54	W	0x0
21	E53	Event #53	W	0x0
20	E52	Event #52	W	0x0
19	E51	Event #51	W	0x0
18	E50	Event #50	W	0x0
17	E49	Event #49	W	0x0
16	E48	Event #48	W	0x0
15	E47	Event #47	W	0x0
14	E46	Event #46	W	0x0
13	E45	Event #45	W	0x0
12	E44	Event #44	W	0x0
11	E43	Event #43	W	0x0
10	E42	Event #42	W	0x0
9	E41	Event #41	W	0x0
8	E40	Event #40	W	0x0
7	E39	Event #39	W	0x0
6	E38	Event #38	W	0x0
5	E37	Event #37	W	0x0
4	E36	Event #36	W	0x0
3	E35	Event #35	W	0x0
2	E34	Event #34	W	0x0
1	E33	Event #33	W	0x0
0	E32	Event #32	W	0x0

Table 16-288. Register Call Summary for Register EDMA_TPCC_EECRH_RN_k

Enhanced DMA

- EDMA Register Summary: [0] [1] [2] [3] [4] [5]

Table 16-289. EDMA_TPCC_EESR_RN_k

Address Offset	0x0000 2030 + (0x200 * k)		
Physical Address	0x4330 2030 + (0x200 * k) 0x40D1 2030 + (0x200 * k) 0x4151 2030 + (0x200 * k) 0x01D1 2030 + (0x200 * k) 0x420A 2030 + (0x200 * k) 0x421A 2030 + (0x200 * k) 0x422A 2030 + (0x200 * k) 0x423A 2030 + (0x200 * k)	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Event Enable Set Register CPU write of '0' has no effect. CPU write of '1' to the EDMA_TPCC_EESR.En bit causes the EDMA_TPCC_EER.En bit to be set.		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E31	E30	E29	E28	E27	E26	E25	E24	E23	E22	E21	E20	E19	E18	E17	E16	E15	E14	E13	E12	E11	E10	E9	E8	E7	E6	E5	E4	E3	E2	E1	E0

Bits	Field Name	Description	Type	Reset
31	E31	Event #31	W	0x0
30	E30	Event #30	W	0x0
29	E29	Event #29	W	0x0
28	E28	Event #28	W	0x0
27	E27	Event #27	W	0x0
26	E26	Event #26	W	0x0
25	E25	Event #25	W	0x0
24	E24	Event #24	W	0x0
23	E23	Event #23	W	0x0
22	E22	Event #22	W	0x0
21	E21	Event #21	W	0x0
20	E20	Event #20	W	0x0
19	E19	Event #19	W	0x0
18	E18	Event #18	W	0x0
17	E17	Event #17	W	0x0
16	E16	Event #16	W	0x0
15	E15	Event #15	W	0x0
14	E14	Event #14	W	0x0
13	E13	Event #13	W	0x0
12	E12	Event #12	W	0x0
11	E11	Event #11	W	0x0
10	E10	Event #10	W	0x0
9	E9	Event #9	W	0x0
8	E8	Event #8	W	0x0
7	E7	Event #7	W	0x0
6	E6	Event #6	W	0x0
5	E5	Event #5	W	0x0

Bits	Field Name	Description	Type	Reset
4	E4	Event #4	W	0x0
3	E3	Event #3	W	0x0
2	E2	Event #2	W	0x0
1	E1	Event #1	W	0x0
0	E0	Event #0	W	0x0

Table 16-290. Register Call Summary for Register EDMA_TPCC_EESR_RN_k

Enhanced DMA

- EDMA Register Summary: [0] [1] [2] [3] [4] [5]

Table 16-291. EDMA_TPCC_EESRH_RN_k

Address Offset	0x0000 2034 + (0x200 * k)		
Physical Address	0x4330 2034 + (0x200 * k) 0x40D1 2034 + (0x200 * k) 0x4151 2034 + (0x200 * k) 0x01D1 2034 + (0x200 * k) 0x420A 2034 + (0x200 * k) 0x421A 2034 + (0x200 * k) 0x422A 2034 + (0x200 * k) 0x423A 2034 + (0x200 * k)	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Event Enable Set Register (High Part) CPU write of '0' has no effect. CPU write of '1' to the EDMA_TPCC_EESRH.En bit causes the EDMA_TPCC_EERH.En bit to be set.		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E63	E62	E61	E60	E59	E58	E57	E56	E55	E54	E53	E52	E51	E50	E49	E48	E47	E46	E45	E44	E43	E42	E41	E40	E39	E38	E37	E36	E35	E34	E33	E32

Bits	Field Name	Description	Type	Reset
31	E63	Event #63	W	0x0
30	E62	Event #62	W	0x0
29	E61	Event #61	W	0x0
28	E60	Event #60	W	0x0
27	E59	Event #59	W	0x0
26	E58	Event #58	W	0x0
25	E57	Event #57	W	0x0
24	E56	Event #56	W	0x0
23	E55	Event #55	W	0x0
22	E54	Event #54	W	0x0
21	E53	Event #53	W	0x0
20	E52	Event #52	W	0x0
19	E51	Event #51	W	0x0
18	E50	Event #50	W	0x0
17	E49	Event #49	W	0x0
16	E48	Event #48	W	0x0
15	E47	Event #47	W	0x0
14	E46	Event #46	W	0x0
13	E45	Event #45	W	0x0
12	E44	Event #44	W	0x0

Bits	Field Name	Description	Type	Reset
11	E43	Event #43	W	0x0
10	E42	Event #42	W	0x0
9	E41	Event #41	W	0x0
8	E40	Event #40	W	0x0
7	E39	Event #39	W	0x0
6	E38	Event #38	W	0x0
5	E37	Event #37	W	0x0
4	E36	Event #36	W	0x0
3	E35	Event #35	W	0x0
2	E34	Event #34	W	0x0
1	E33	Event #33	W	0x0
0	E32	Event #32	W	0x0

Table 16-292. Register Call Summary for Register EDMA_TPCC_EESRH_RN_k

Enhanced DMA
- EDMA Register Summary: [0] [1] [2] [3] [4] [5]

Table 16-293. EDMA_TPCC_SER_RN_k

Address Offset	0x0000 2038 + (0x200 * k)		
Physical Address	0x4330 2038 + (0x200 * k) 0x40D1 2038 + (0x200 * k) 0x4151 2038 + (0x200 * k) 0x01D1 2038 + (0x200 * k) 0x420A 2038 + (0x200 * k) 0x421A 2038 + (0x200 * k) 0x422A 2038 + (0x200 * k) 0x423A 2038 + (0x200 * k)	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Secondary Event Register The secondary event register is used along with the Event Register (EDMA_TPCC_ER) to provide information on the state of an Event. En = 0 : Event is not currently in the Event Queue. En = 1 : Event is currently stored in Event Queue. Event arbiter will not prioritize additional events.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E31	E30	E29	E28	E27	E26	E25	E24	E23	E22	E21	E20	E19	E18	E17	E16	E15	E14	E13	E12	E11	E10	E9	E8	E7	E6	E5	E4	E3	E2	E1	E0

Bits	Field Name	Description	Type	Reset
31	E31	Event #31	R	0x0
30	E30	Event #30	R	0x0
29	E29	Event #29	R	0x0
28	E28	Event #28	R	0x0
27	E27	Event #27	R	0x0
26	E26	Event #26	R	0x0
25	E25	Event #25	R	0x0
24	E24	Event #24	R	0x0
23	E23	Event #23	R	0x0
22	E22	Event #22	R	0x0
21	E21	Event #21	R	0x0
20	E20	Event #20	R	0x0

Bits	Field Name	Description	Type	Reset
19	E19	Event #19	R	0x0
18	E18	Event #18	R	0x0
17	E17	Event #17	R	0x0
16	E16	Event #16	R	0x0
15	E15	Event #15	R	0x0
14	E14	Event #14	R	0x0
13	E13	Event #13	R	0x0
12	E12	Event #12	R	0x0
11	E11	Event #11	R	0x0
10	E10	Event #10	R	0x0
9	E9	Event #9	R	0x0
8	E8	Event #8	R	0x0
7	E7	Event #7	R	0x0
6	E6	Event #6	R	0x0
5	E5	Event #5	R	0x0
4	E4	Event #4	R	0x0
3	E3	Event #3	R	0x0
2	E2	Event #2	R	0x0
1	E1	Event #1	R	0x0
0	E0	Event #0	R	0x0

Table 16-294. Register Call Summary for Register EDMA_TPCC_SER_RN_k

Enhanced DMA

- EDMA Register Summary: [0] [1] [2] [3] [4] [5]

Table 16-295. EDMA_TPCC_SERH_RN_k

Address Offset	0x0000 203C + (0x200 * k)		
Physical Address	0x4330 203C + (0x200 * k) 0x40D1 203C + (0x200 * k) 0x4151 203C + (0x200 * k) 0x01D1 203C + (0x200 * k) 0x420A 203C + (0x200 * k) 0x421A 203C + (0x200 * k) 0x422A 203C + (0x200 * k) 0x423A 203C + (0x200 * k)	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Secondary Event Register (High Part) The secondary event register is used along with the Event Register (EDMA_TPCC_ERH) to provide information on the state of an Event. En = 0 : Event is not currently in the Event Queue. En = 1 : Event is currently stored in Event Queue. Event arbiter will not prioritize additional events.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E63	E62	E61	E60	E59	E58	E57	E56	E55	E54	E53	E52	E51	E50	E49	E48	E47	E46	E45	E44	E43	E42	E41	E40	E39	E38	E37	E36	E35	E34	E33	E32

Bits	Field Name	Description	Type	Reset
31	E63	Event #63	R	0x0
30	E62	Event #62	R	0x0
29	E61	Event #61	R	0x0
28	E60	Event #60	R	0x0

Bits	Field Name	Description	Type	Reset
27	E59	Event #59	R	0x0
26	E58	Event #58	R	0x0
25	E57	Event #57	R	0x0
24	E56	Event #56	R	0x0
23	E55	Event #55	R	0x0
22	E54	Event #54	R	0x0
21	E53	Event #53	R	0x0
20	E52	Event #52	R	0x0
19	E51	Event #51	R	0x0
18	E50	Event #50	R	0x0
17	E49	Event #49	R	0x0
16	E48	Event #48	R	0x0
15	E47	Event #47	R	0x0
14	E46	Event #46	R	0x0
13	E45	Event #45	R	0x0
12	E44	Event #44	R	0x0
11	E43	Event #43	R	0x0
10	E42	Event #42	R	0x0
9	E41	Event #41	R	0x0
8	E40	Event #40	R	0x0
7	E39	Event #39	R	0x0
6	E38	Event #38	R	0x0
5	E37	Event #37	R	0x0
4	E36	Event #36	R	0x0
3	E35	Event #35	R	0x0
2	E34	Event #34	R	0x0
1	E33	Event #33	R	0x0
0	E32	Event #32	R	0x0

Table 16-296. Register Call Summary for Register EDMA_TPCC_SERH_RN_k

Enhanced DMA
- EDMA Register Summary: [0] [1] [2] [3] [4] [5]

Table 16-297. EDMA_TPCC_SECR_RN_k

Address Offset	0x0000 2040 + (0x200 * k)		
Physical Address	0x4330 2040 + (0x200 * k) 0x40D1 2040 + (0x200 * k) 0x4151 2040 + (0x200 * k) 0x01D1 2040 + (0x200 * k) 0x420A 2040 + (0x200 * k) 0x421A 2040 + (0x200 * k) 0x422A 2040 + (0x200 * k) 0x423A 2040 + (0x200 * k)	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Secondary Event Clear Register The secondary event clear register is used to clear the status of the EDMA_TPCC_SER registers. CPU write of '0' has no effect. CPU write of '1' to the EDMA_TPCC_SECR.En bit clears the EDMA_TPCC_SER register.		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E31	E30	E29	E28	E27	E26	E25	E24	E23	E22	E21	E20	E19	E18	E17	E16	E15	E14	E13	E12	E11	E10	E9	E8	E7	E6	E5	E4	E3	E2	E1	E0

Bits	Field Name	Description	Type	Reset
31	E31	Event #31	W	0x0
30	E30	Event #30	W	0x0
29	E29	Event #29	W	0x0
28	E28	Event #28	W	0x0
27	E27	Event #27	W	0x0
26	E26	Event #26	W	0x0
25	E25	Event #25	W	0x0
24	E24	Event #24	W	0x0
23	E23	Event #23	W	0x0
22	E22	Event #22	W	0x0
21	E21	Event #21	W	0x0
20	E20	Event #20	W	0x0
19	E19	Event #19	W	0x0
18	E18	Event #18	W	0x0
17	E17	Event #17	W	0x0
16	E16	Event #16	W	0x0
15	E15	Event #15	W	0x0
14	E14	Event #14	W	0x0
13	E13	Event #13	W	0x0
12	E12	Event #12	W	0x0
11	E11	Event #11	W	0x0
10	E10	Event #10	W	0x0
9	E9	Event #9	W	0x0
8	E8	Event #8	W	0x0
7	E7	Event #7	W	0x0
6	E6	Event #6	W	0x0
5	E5	Event #5	W	0x0
4	E4	Event #4	W	0x0
3	E3	Event #3	W	0x0
2	E2	Event #2	W	0x0
1	E1	Event #1	W	0x0
0	E0	Event #0	W	0x0

Table 16-298. Register Call Summary for Register EDMA_TPCC_SECR_RN_k

Enhanced DMA

- EDMA Register Summary: [0] [1] [2] [3] [4] [5]

Table 16-299. EDMA_TPCC_SECRH_RN_k

Address Offset	0x0000 2044 + (0x200 * k)		
Physical Address	0x4330 2044 + (0x200 * k) 0x40D1 2044 + (0x200 * k) 0x4151 2044 + (0x200 * k) 0x01D1 2044 + (0x200 * k) 0x420A 2044 + (0x200 * k) 0x421A 2044 + (0x200 * k) 0x422A 2044 + (0x200 * k) 0x423A 2044 + (0x200 * k)	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC

Table 16-299. EDMA_TPCC_SECRH_RN_k (continued)

Description	Secondary Event Clear Register (High Part) The secondary event clear register is used to clear the status of the EDMA_TPCC_SERH registers. CPU write of '0' has no effect. CPU write of '1' to the EDMA_TPCC_SECRH.En bit clears the EDMA_TPCC_SERH register.
Type	W

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E63	E62	E61	E60	E59	E58	E57	E56	E55	E54	E53	E52	E51	E50	E49	E48	E47	E46	E45	E44	E43	E42	E41	E40	E39	E38	E37	E36	E35	E34	E33	E32

Bits	Field Name	Description	Type	Reset
31	E63	Event #63	W	0x0
30	E62	Event #62	W	0x0
29	E61	Event #61	W	0x0
28	E60	Event #60	W	0x0
27	E59	Event #59	W	0x0
26	E58	Event #58	W	0x0
25	E57	Event #57	W	0x0
24	E56	Event #56	W	0x0
23	E55	Event #55	W	0x0
22	E54	Event #54	W	0x0
21	E53	Event #53	W	0x0
20	E52	Event #52	W	0x0
19	E51	Event #51	W	0x0
18	E50	Event #50	W	0x0
17	E49	Event #49	W	0x0
16	E48	Event #48	W	0x0
15	E47	Event #47	W	0x0
14	E46	Event #46	W	0x0
13	E45	Event #45	W	0x0
12	E44	Event #44	W	0x0
11	E43	Event #43	W	0x0
10	E42	Event #42	W	0x0
9	E41	Event #41	W	0x0
8	E40	Event #40	W	0x0
7	E39	Event #39	W	0x0
6	E38	Event #38	W	0x0
5	E37	Event #37	W	0x0
4	E36	Event #36	W	0x0
3	E35	Event #35	W	0x0
2	E34	Event #34	W	0x0
1	E33	Event #33	W	0x0
0	E32	Event #32	W	0x0

Table 16-300. Register Call Summary for Register EDMA_TPCC_SECRH_RN_k

Enhanced DMA

- EDMA Register Summary: [0] [1] [2] [3] [4] [5]

Table 16-301. EDMA_TPCC_IER_RN_k

Address Offset	0x0000 2050 + (0x200 * k)		
Physical Address	0x4330 2050 + (0x200 * k) 0x40D1 2050 + (0x200 * k) 0x4151 2050 + (0x200 * k) 0x01D1 2050 + (0x200 * k) 0x420A 2050 + (0x200 * k) 0x421A 2050 + (0x200 * k) 0x422A 2050 + (0x200 * k) 0x423A 2050 + (0x200 * k)	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Int Enable Register EDMA_TPCC_IER.In is not directly writeable. Interrupts can be enabled via writes to EDMA_TPCC_IESR and can be disabled via writes to EDMA_TPCC_IECR register. EDMA_TPCC_IER.In = 0: EDMA_TPCC_IPR.In is NOT enabled for interrupts. EDMA_TPCC_IER.In = 1: EDMA_TPCC_IPR.In IS enabled for interrupts.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
I31	I30	I29	I28	I27	I26	I25	I24	I23	I22	I21	I20	I19	I18	I17	I16	I15	I14	I13	I12	I11	I10	I9	I8	I7	I6	I5	I4	I3	I2	I1	I0

Bits	Field Name	Description	Type	Reset
31	I31	Interrupt associated with TCC #31	R	0x0
30	I30	Interrupt associated with TCC #30	R	0x0
29	I29	Interrupt associated with TCC #29	R	0x0
28	I28	Interrupt associated with TCC #28	R	0x0
27	I27	Interrupt associated with TCC #27	R	0x0
26	I26	Interrupt associated with TCC #26	R	0x0
25	I25	Interrupt associated with TCC #25	R	0x0
24	I24	Interrupt associated with TCC #24	R	0x0
23	I23	Interrupt associated with TCC #23	R	0x0
22	I22	Interrupt associated with TCC #22	R	0x0
21	I21	Interrupt associated with TCC #21	R	0x0
20	I20	Interrupt associated with TCC #20	R	0x0
19	I19	Interrupt associated with TCC #19	R	0x0
18	I18	Interrupt associated with TCC #18	R	0x0
17	I17	Interrupt associated with TCC #17	R	0x0
16	I16	Interrupt associated with TCC #16	R	0x0
15	I15	Interrupt associated with TCC #15	R	0x0
14	I14	Interrupt associated with TCC #14	R	0x0
13	I13	Interrupt associated with TCC #13	R	0x0
12	I12	Interrupt associated with TCC #12	R	0x0
11	I11	Interrupt associated with TCC #11	R	0x0
10	I10	Interrupt associated with TCC #10	R	0x0
9	I9	Interrupt associated with TCC #9	R	0x0
8	I8	Interrupt associated with TCC #8	R	0x0
7	I7	Interrupt associated with TCC #7	R	0x0
6	I6	Interrupt associated with TCC #6	R	0x0
5	I5	Interrupt associated with TCC #5	R	0x0
4	I4	Interrupt associated with TCC #4	R	0x0
3	I3	Interrupt associated with TCC #3	R	0x0
2	I2	Interrupt associated with TCC #2	R	0x0
1	I1	Interrupt associated with TCC #1	R	0x0

Bits	Field Name	Description	Type	Reset
0	I0	Interrupt associated with TCC #0	R	0x0

Table 16-302. Register Call Summary for Register EDMA_TPCC_IER_RN_k

Enhanced DMA

- EDMA Register Summary: [0] [1] [2] [3] [4] [5]

Table 16-303. EDMA_TPCC_IERH_RN_k

Address Offset	0x0000 2054 + (0x200 * k)		
Physical Address	0x4330 2054 + (0x200 * k) 0x40D1 2054 + (0x200 * k) 0x4151 2054 + (0x200 * k) 0x01D1 2054 + (0x200 * k) 0x420A 2054 + (0x200 * k) 0x421A 2054 + (0x200 * k) 0x422A 2054 + (0x200 * k) 0x423A 2054 + (0x200 * k)	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Int Enable Register (High Part) EDMA_TPCC_IERH.In is not directly writeable. Interrupts can be enabled via writes to EDMA_TPCC_IESRH and can be disabled via writes to EDMA_TPCC_IECRH register. EDMA_TPCC_IERH.In = 0: EDMA_TPCC_IPRH.In is NOT enabled for interrupts. EDMA_TPCC_IERH.In = 1: EDMA_TPCC_IPRH.In IS enabled for interrupts.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
I63	I62	I61	I60	I59	I58	I57	I56	I55	I54	I53	I52	I51	I50	I49	I48	I47	I46	I45	I44	I43	I42	I41	I40	I39	I38	I37	I36	I35	I34	I33	I32

Bits	Field Name	Description	Type	Reset
31	I63	Interrupt associated with TCC #63	R	0x0
30	I62	Interrupt associated with TCC #62	R	0x0
29	I61	Interrupt associated with TCC #61	R	0x0
28	I60	Interrupt associated with TCC #60	R	0x0
27	I59	Interrupt associated with TCC #59	R	0x0
26	I58	Interrupt associated with TCC #58	R	0x0
25	I57	Interrupt associated with TCC #57	R	0x0
24	I56	Interrupt associated with TCC #56	R	0x0
23	I55	Interrupt associated with TCC #55	R	0x0
22	I54	Interrupt associated with TCC #54	R	0x0
21	I53	Interrupt associated with TCC #53	R	0x0
20	I52	Interrupt associated with TCC #52	R	0x0
19	I51	Interrupt associated with TCC #51	R	0x0
18	I50	Interrupt associated with TCC #50	R	0x0
17	I49	Interrupt associated with TCC #49	R	0x0
16	I48	Interrupt associated with TCC #48	R	0x0
15	I47	Interrupt associated with TCC #47	R	0x0
14	I46	Interrupt associated with TCC #46	R	0x0
13	I45	Interrupt associated with TCC #45	R	0x0
12	I44	Interrupt associated with TCC #44	R	0x0
11	I43	Interrupt associated with TCC #43	R	0x0
10	I42	Interrupt associated with TCC #42	R	0x0
9	I41	Interrupt associated with TCC #41	R	0x0
8	I40	Interrupt associated with TCC #40	R	0x0

Bits	Field Name	Description	Type	Reset
7	I39	Interrupt associated with TCC #39	R	0x0
6	I38	Interrupt associated with TCC #38	R	0x0
5	I37	Interrupt associated with TCC #37	R	0x0
4	I36	Interrupt associated with TCC #36	R	0x0
3	I35	Interrupt associated with TCC #35	R	0x0
2	I34	Interrupt associated with TCC #34	R	0x0
1	I33	Interrupt associated with TCC #33	R	0x0
0	I32	Interrupt associated with TCC #32	R	0x0

Table 16-304. Register Call Summary for Register EDMA_TPCC_IERH_RN_k

Enhanced DMA
- EDMA Register Summary: [0] [1] [2] [3] [4] [5]

Table 16-305. EDMA_TPCC_IECR_RN_k

Address Offset	0x0000 2058 + (0x200 * k)		
Physical Address	0x4330 2058 + (0x200 * k) 0x40D1 2058 + (0x200 * k) 0x4151 2058 + (0x200 * k) 0x01D1 2058 + (0x200 * k) 0x420A 2058 + (0x200 * k) 0x421A 2058 + (0x200 * k) 0x422A 2058 + (0x200 * k) 0x423A 2058 + (0x200 * k)	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Int Enable Clear Register CPU write of '0' has no effect. CPU write of '1' to the EDMA_TPCC_IECR.In bit causes the EDMA_TPCC_IER.In bit to be cleared.		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
I31	I30	I29	I28	I27	I26	I25	I24	I23	I22	I21	I20	I19	I18	I17	I16	I15	I14	I13	I12	I11	I10	I9	I8	I7	I6	I5	I4	I3	I2	I1	I0

Bits	Field Name	Description	Type	Reset
31	I31	Interrupt associated with TCC #31	W	0x0
30	I30	Interrupt associated with TCC #30	W	0x0
29	I29	Interrupt associated with TCC #29	W	0x0
28	I28	Interrupt associated with TCC #28	W	0x0
27	I27	Interrupt associated with TCC #27	W	0x0
26	I26	Interrupt associated with TCC #26	W	0x0
25	I25	Interrupt associated with TCC #25	W	0x0
24	I24	Interrupt associated with TCC #24	W	0x0
23	I23	Interrupt associated with TCC #23	W	0x0
22	I22	Interrupt associated with TCC #22	W	0x0
21	I21	Interrupt associated with TCC #21	W	0x0
20	I20	Interrupt associated with TCC #20	W	0x0
19	I19	Interrupt associated with TCC #19	W	0x0
18	I18	Interrupt associated with TCC #18	W	0x0
17	I17	Interrupt associated with TCC #17	W	0x0
16	I16	Interrupt associated with TCC #16	W	0x0
15	I15	Interrupt associated with TCC #15	W	0x0

Bits	Field Name	Description	Type	Reset
14	I14	Interrupt associated with TCC #14	W	0x0
13	I13	Interrupt associated with TCC #13	W	0x0
12	I12	Interrupt associated with TCC #12	W	0x0
11	I11	Interrupt associated with TCC #11	W	0x0
10	I10	Interrupt associated with TCC #10	W	0x0
9	I9	Interrupt associated with TCC #9	W	0x0
8	I8	Interrupt associated with TCC #8	W	0x0
7	I7	Interrupt associated with TCC #7	W	0x0
6	I6	Interrupt associated with TCC #6	W	0x0
5	I5	Interrupt associated with TCC #5	W	0x0
4	I4	Interrupt associated with TCC #4	W	0x0
3	I3	Interrupt associated with TCC #3	W	0x0
2	I2	Interrupt associated with TCC #2	W	0x0
1	I1	Interrupt associated with TCC #1	W	0x0
0	I0	Interrupt associated with TCC #0	W	0x0

Table 16-306. Register Call Summary for Register EDMA_TPCC_IECR_RN_k

Enhanced DMA

- EDMA Register Summary: [0] [1] [2] [3] [4] [5]

Table 16-307. EDMA_TPCC_IECRH_RN_k

Address Offset	0x0000 205C + (0x200 * k)		
Physical Address	0x4330 205C + (0x200 * k) 0x40D1 205C + (0x200 * k) 0x4151 205C + (0x200 * k) 0x01D1 205C + (0x200 * k) 0x420A 205C + (0x200 * k) 0x421A 205C + (0x200 * k) 0x422A 205C + (0x200 * k) 0x423A 205C + (0x200 * k)	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Int Enable Clear Register (High Part) CPU write of '0' has no effect. CPU write of '1' to the EDMA_TPCC_IECRH.In bit causes the EDMA_TPCC_IERH.In bit to be cleared.		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
I63	I62	I61	I60	I59	I58	I57	I56	I55	I54	I53	I52	I51	I50	I49	I48	I47	I46	I45	I44	I43	I42	I41	I40	I39	I38	I37	I36	I35	I34	I33	I32

Bits	Field Name	Description	Type	Reset
31	I63	Interrupt associated with TCC #63	W	0x0
30	I62	Interrupt associated with TCC #62	W	0x0
29	I61	Interrupt associated with TCC #61	W	0x0
28	I60	Interrupt associated with TCC #60	W	0x0
27	I59	Interrupt associated with TCC #59	W	0x0
26	I58	Interrupt associated with TCC #58	W	0x0
25	I57	Interrupt associated with TCC #57	W	0x0
24	I56	Interrupt associated with TCC #56	W	0x0
23	I55	Interrupt associated with TCC #55	W	0x0
22	I54	Interrupt associated with TCC #54	W	0x0

Bits	Field Name	Description	Type	Reset
21	I53	Interrupt associated with TCC #53	W	0x0
20	I52	Interrupt associated with TCC #52	W	0x0
19	I51	Interrupt associated with TCC #51	W	0x0
18	I50	Interrupt associated with TCC #50	W	0x0
17	I49	Interrupt associated with TCC #49	W	0x0
16	I48	Interrupt associated with TCC #48	W	0x0
15	I47	Interrupt associated with TCC #47	W	0x0
14	I46	Interrupt associated with TCC #46	W	0x0
13	I45	Interrupt associated with TCC #45	W	0x0
12	I44	Interrupt associated with TCC #44	W	0x0
11	I43	Interrupt associated with TCC #43	W	0x0
10	I42	Interrupt associated with TCC #42	W	0x0
9	I41	Interrupt associated with TCC #41	W	0x0
8	I40	Interrupt associated with TCC #40	W	0x0
7	I39	Interrupt associated with TCC #39	W	0x0
6	I38	Interrupt associated with TCC #38	W	0x0
5	I37	Interrupt associated with TCC #37	W	0x0
4	I36	Interrupt associated with TCC #36	W	0x0
3	I35	Interrupt associated with TCC #35	W	0x0
2	I34	Interrupt associated with TCC #34	W	0x0
1	I33	Interrupt associated with TCC #33	W	0x0
0	I32	Interrupt associated with TCC #32	W	0x0

Table 16-308. Register Call Summary for Register EDMA_TPCC_IECRH_RN_k

Enhanced DMA

- EDMA Register Summary: [0] [1] [2] [3] [4] [5]

Table 16-309. EDMA_TPCC_IESR_RN_k

Address Offset	0x0000 2060 + (0x200 * k)		
Physical Address	0x4330 2060 + (0x200 * k) 0x40D1 2060 + (0x200 * k) 0x4151 2060 + (0x200 * k) 0x01D1 2060 + (0x200 * k) 0x420A 2060 + (0x200 * k) 0x421A 2060 + (0x200 * k) 0x422A 2060 + (0x200 * k) 0x423A 2060 + (0x200 * k)	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Int Enable Set Register CPU write of '0' has no effect. CPU write of '1' to the EDMA_TPCC_IESR.In bit causes the EDMA_TPCC_IESR.In bit to be set.		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
I31	I30	I29	I28	I27	I26	I25	I24	I23	I22	I21	I20	I19	I18	I17	I16	I15	I14	I13	I12	I11	I10	I9	I8	I7	I6	I5	I4	I3	I2	I1	I0

Bits	Field Name	Description	Type	Reset
31	I31	Interrupt associated with TCC #31	W	0x0
30	I30	Interrupt associated with TCC #30	W	0x0
29	I29	Interrupt associated with TCC #29	W	0x0
28	I28	Interrupt associated with TCC #28	W	0x0

Bits	Field Name	Description	Type	Reset
27	I27	Interrupt associated with TCC #27	W	0x0
26	I26	Interrupt associated with TCC #26	W	0x0
25	I25	Interrupt associated with TCC #25	W	0x0
24	I24	Interrupt associated with TCC #24	W	0x0
23	I23	Interrupt associated with TCC #23	W	0x0
22	I22	Interrupt associated with TCC #22	W	0x0
21	I21	Interrupt associated with TCC #21	W	0x0
20	I20	Interrupt associated with TCC #20	W	0x0
19	I19	Interrupt associated with TCC #19	W	0x0
18	I18	Interrupt associated with TCC #18	W	0x0
17	I17	Interrupt associated with TCC #17	W	0x0
16	I16	Interrupt associated with TCC #16	W	0x0
15	I15	Interrupt associated with TCC #15	W	0x0
14	I14	Interrupt associated with TCC #14	W	0x0
13	I13	Interrupt associated with TCC #13	W	0x0
12	I12	Interrupt associated with TCC #12	W	0x0
11	I11	Interrupt associated with TCC #11	W	0x0
10	I10	Interrupt associated with TCC #10	W	0x0
9	I9	Interrupt associated with TCC #9	W	0x0
8	I8	Interrupt associated with TCC #8	W	0x0
7	I7	Interrupt associated with TCC #7	W	0x0
6	I6	Interrupt associated with TCC #6	W	0x0
5	I5	Interrupt associated with TCC #5	W	0x0
4	I4	Interrupt associated with TCC #4	W	0x0
3	I3	Interrupt associated with TCC #3	W	0x0
2	I2	Interrupt associated with TCC #2	W	0x0
1	I1	Interrupt associated with TCC #1	W	0x0
0	I0	Interrupt associated with TCC #0	W	0x0

Table 16-310. Register Call Summary for Register EDMA_TPCC_IESR_RN_k

Enhanced DMA
- EDMA Register Summary: [0] [1] [2] [3] [4] [5]

Table 16-311. EDMA_TPCC_IESRH_RN_k

Address Offset	0x0000 2064 + (0x200 * k)		
Physical Address	0x4330 2064 + (0x200 * k) 0x40D1 2064 + (0x200 * k) 0x4151 2064 + (0x200 * k) 0x01D1 2064 + (0x200 * k) 0x420A 2064 + (0x200 * k) 0x421A 2064 + (0x200 * k) 0x422A 2064 + (0x200 * k) 0x423A 2064 + (0x200 * k)	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Int Enable Set Register (High Part) CPU write of '0' has no effect. CPU write of '1' to the EDMA_TPCC_IESRH.In bit causes the EDMA_TPCC_IESRH.In bit to be set.		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
I63	I62	I61	I60	I59	I58	I57	I56	I55	I54	I53	I52	I51	I50	I49	I48	I47	I46	I45	I44	I43	I42	I41	I40	I39	I38	I37	I36	I35	I34	I33	I32

Bits	Field Name	Description	Type	Reset
31	I63	Interrupt associated with TCC #63	W	0x0
30	I62	Interrupt associated with TCC #62	W	0x0
29	I61	Interrupt associated with TCC #61	W	0x0
28	I60	Interrupt associated with TCC #60	W	0x0
27	I59	Interrupt associated with TCC #59	W	0x0
26	I58	Interrupt associated with TCC #58	W	0x0
25	I57	Interrupt associated with TCC #57	W	0x0
24	I56	Interrupt associated with TCC #56	W	0x0
23	I55	Interrupt associated with TCC #55	W	0x0
22	I54	Interrupt associated with TCC #54	W	0x0
21	I53	Interrupt associated with TCC #53	W	0x0
20	I52	Interrupt associated with TCC #52	W	0x0
19	I51	Interrupt associated with TCC #51	W	0x0
18	I50	Interrupt associated with TCC #50	W	0x0
17	I49	Interrupt associated with TCC #49	W	0x0
16	I48	Interrupt associated with TCC #48	W	0x0
15	I47	Interrupt associated with TCC #47	W	0x0
14	I46	Interrupt associated with TCC #46	W	0x0
13	I45	Interrupt associated with TCC #45	W	0x0
12	I44	Interrupt associated with TCC #44	W	0x0
11	I43	Interrupt associated with TCC #43	W	0x0
10	I42	Interrupt associated with TCC #42	W	0x0
9	I41	Interrupt associated with TCC #41	W	0x0
8	I40	Interrupt associated with TCC #40	W	0x0
7	I39	Interrupt associated with TCC #39	W	0x0
6	I38	Interrupt associated with TCC #38	W	0x0
5	I37	Interrupt associated with TCC #37	W	0x0
4	I36	Interrupt associated with TCC #36	W	0x0
3	I35	Interrupt associated with TCC #35	W	0x0
2	I34	Interrupt associated with TCC #34	W	0x0
1	I33	Interrupt associated with TCC #33	W	0x0
0	I32	Interrupt associated with TCC #32	W	0x0

Table 16-312. Register Call Summary for Register EDMA_TPCC_IESRH_RN_k

Enhanced DMA

- EDMA Register Summary: [0] [1] [2] [3] [4] [5]

Table 16-313. EDMA_TPCC_IPR_RN_k

Address Offset	0x0000 2068 + (0x200 * k)		
Physical Address	0x4330 2068 + (0x200 * k) 0x40D1 2068 + (0x200 * k) 0x4151 2068 + (0x200 * k) 0x01D1 2068 + (0x200 * k) 0x420A 2068 + (0x200 * k) 0x421A 2068 + (0x200 * k) 0x422A 2068 + (0x200 * k) 0x423A 2068 + (0x200 * k)	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC

Table 16-313. EDMA_TPCC_IPR_RN_k (continued)

Description	Interrupt Pending Register EDMA_TPCC_IPR.In bit is set when a interrupt completion code with TCC of N is detected. EDMA_TPCC_IPR.In bit is cleared via software by writing a '1' to EDMA_TPCC_ICR.In bit.
Type	R

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
I31	I30	I29	I28	I27	I26	I25	I24	I23	I22	I21	I20	I19	I18	I17	I16	I15	I14	I13	I12	I11	I10	I9	I8	I7	I6	I5	I4	I3	I2	I1	I0

Bits	Field Name	Description	Type	Reset
31	I31	Interrupt associated with TCC #31	R	0x0
30	I30	Interrupt associated with TCC #30	R	0x0
29	I29	Interrupt associated with TCC #29	R	0x0
28	I28	Interrupt associated with TCC #28	R	0x0
27	I27	Interrupt associated with TCC #27	R	0x0
26	I26	Interrupt associated with TCC #26	R	0x0
25	I25	Interrupt associated with TCC #25	R	0x0
24	I24	Interrupt associated with TCC #24	R	0x0
23	I23	Interrupt associated with TCC #23	R	0x0
22	I22	Interrupt associated with TCC #22	R	0x0
21	I21	Interrupt associated with TCC #21	R	0x0
20	I20	Interrupt associated with TCC #20	R	0x0
19	I19	Interrupt associated with TCC #19	R	0x0
18	I18	Interrupt associated with TCC #18	R	0x0
17	I17	Interrupt associated with TCC #17	R	0x0
16	I16	Interrupt associated with TCC #16	R	0x0
15	I15	Interrupt associated with TCC #15	R	0x0
14	I14	Interrupt associated with TCC #14	R	0x0
13	I13	Interrupt associated with TCC #13	R	0x0
12	I12	Interrupt associated with TCC #12	R	0x0
11	I11	Interrupt associated with TCC #11	R	0x0
10	I10	Interrupt associated with TCC #10	R	0x0
9	I9	Interrupt associated with TCC #9	R	0x0
8	I8	Interrupt associated with TCC #8	R	0x0
7	I7	Interrupt associated with TCC #7	R	0x0
6	I6	Interrupt associated with TCC #6	R	0x0
5	I5	Interrupt associated with TCC #5	R	0x0
4	I4	Interrupt associated with TCC #4	R	0x0
3	I3	Interrupt associated with TCC #3	R	0x0
2	I2	Interrupt associated with TCC #2	R	0x0
1	I1	Interrupt associated with TCC #1	R	0x0
0	I0	Interrupt associated with TCC #0	R	0x0

Table 16-314. Register Call Summary for Register EDMA_TPCC_IPR_RN_k

Enhanced DMA

- EDMA Register Summary: [0] [1] [2] [3] [4] [5]

Table 16-315. EDMA_TPCC_IPRH_RN_k

Address Offset	0x0000 206C + (0x200 * k)		
Physical Address	0x4330 206C + (0x200 * k) 0x40D1 206C + (0x200 * k) 0x4151 206C + (0x200 * k) 0x01D1 206C + (0x200 * k) 0x420A 206C + (0x200 * k) 0x421A 206C + (0x200 * k) 0x422A 206C + (0x200 * k) 0x423A 206C + (0x200 * k)	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Interrupt Pending Register (High Part) EDMA_TPCC_IPRH.In bit is set when a interrupt completion code with TCC of N is detected. EDMA_TPCC_IPRH.In bit is cleared via software by writing a '1' to EDMA_TPCC_ICRH.In bit.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
I63	I62	I61	I60	I59	I58	I57	I56	I55	I54	I53	I52	I51	I50	I49	I48	I47	I46	I45	I44	I43	I42	I41	I40	I39	I38	I37	I36	I35	I34	I33	I32

Bits	Field Name	Description	Type	Reset
31	I63	Interrupt associated with TCC #63	R	0x0
30	I62	Interrupt associated with TCC #62	R	0x0
29	I61	Interrupt associated with TCC #61	R	0x0
28	I60	Interrupt associated with TCC #60	R	0x0
27	I59	Interrupt associated with TCC #59	R	0x0
26	I58	Interrupt associated with TCC #58	R	0x0
25	I57	Interrupt associated with TCC #57	R	0x0
24	I56	Interrupt associated with TCC #56	R	0x0
23	I55	Interrupt associated with TCC #55	R	0x0
22	I54	Interrupt associated with TCC #54	R	0x0
21	I53	Interrupt associated with TCC #53	R	0x0
20	I52	Interrupt associated with TCC #52	R	0x0
19	I51	Interrupt associated with TCC #51	R	0x0
18	I50	Interrupt associated with TCC #50	R	0x0
17	I49	Interrupt associated with TCC #49	R	0x0
16	I48	Interrupt associated with TCC #48	R	0x0
15	I47	Interrupt associated with TCC #47	R	0x0
14	I46	Interrupt associated with TCC #46	R	0x0
13	I45	Interrupt associated with TCC #45	R	0x0
12	I44	Interrupt associated with TCC #44	R	0x0
11	I43	Interrupt associated with TCC #43	R	0x0
10	I42	Interrupt associated with TCC #42	R	0x0
9	I41	Interrupt associated with TCC #41	R	0x0
8	I40	Interrupt associated with TCC #40	R	0x0
7	I39	Interrupt associated with TCC #39	R	0x0
6	I38	Interrupt associated with TCC #38	R	0x0
5	I37	Interrupt associated with TCC #37	R	0x0
4	I36	Interrupt associated with TCC #36	R	0x0
3	I35	Interrupt associated with TCC #35	R	0x0
2	I34	Interrupt associated with TCC #34	R	0x0
1	I33	Interrupt associated with TCC #33	R	0x0
0	I32	Interrupt associated with TCC #32	R	0x0

Table 16-316. Register Call Summary for Register EDMA_TPCC_IPRH_RN_k

Enhanced DMA
- EDMA Register Summary: [0] [1] [2] [3] [4] [5]

Table 16-317. EDMA_TPCC_ICR_RN_k

Address Offset	0x0000 2070 + (0x200 * k)		
Physical Address	0x4330 2070 + (0x200 * k) 0x40D1 2070 + (0x200 * k) 0x4151 2070 + (0x200 * k) 0x01D1 2070 + (0x200 * k) 0x420A 2070 + (0x200 * k) 0x421A 2070 + (0x200 * k) 0x422A 2070 + (0x200 * k) 0x423A 2070 + (0x200 * k)	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Interrupt Clear Register CPU writes of '0' has no effect. CPU write of '1' to the EDMA_TPCC_ICR.In bit causes the EDMA_TPCC_IPR.In bit to be cleared. All EDMA_TPCC_IPR.In bits must be cleared before additional interrupts will be asserted by CC.		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
I31	I30	I29	I28	I27	I26	I25	I24	I23	I22	I21	I20	I19	I18	I17	I16	I15	I14	I13	I12	I11	I10	I9	I8	I7	I6	I5	I4	I3	I2	I1	I0

Bits	Field Name	Description	Type	Reset
31	I31	Interrupt associated with TCC #31	W	0x0
30	I30	Interrupt associated with TCC #30	W	0x0
29	I29	Interrupt associated with TCC #29	W	0x0
28	I28	Interrupt associated with TCC #28	W	0x0
27	I27	Interrupt associated with TCC #27	W	0x0
26	I26	Interrupt associated with TCC #26	W	0x0
25	I25	Interrupt associated with TCC #25	W	0x0
24	I24	Interrupt associated with TCC #24	W	0x0
23	I23	Interrupt associated with TCC #23	W	0x0
22	I22	Interrupt associated with TCC #22	W	0x0
21	I21	Interrupt associated with TCC #21	W	0x0
20	I20	Interrupt associated with TCC #20	W	0x0
19	I19	Interrupt associated with TCC #19	W	0x0
18	I18	Interrupt associated with TCC #18	W	0x0
17	I17	Interrupt associated with TCC #17	W	0x0
16	I16	Interrupt associated with TCC #16	W	0x0
15	I15	Interrupt associated with TCC #15	W	0x0
14	I14	Interrupt associated with TCC #14	W	0x0
13	I13	Interrupt associated with TCC #13	W	0x0
12	I12	Interrupt associated with TCC #12	W	0x0
11	I11	Interrupt associated with TCC #11	W	0x0
10	I10	Interrupt associated with TCC #10	W	0x0
9	I9	Interrupt associated with TCC #9	W	0x0
8	I8	Interrupt associated with TCC #8	W	0x0
7	I7	Interrupt associated with TCC #7	W	0x0
6	I6	Interrupt associated with TCC #6	W	0x0
5	I5	Interrupt associated with TCC #5	W	0x0

Bits	Field Name	Description	Type	Reset
4	I4	Interrupt associated with TCC #4	W	0x0
3	I3	Interrupt associated with TCC #3	W	0x0
2	I2	Interrupt associated with TCC #2	W	0x0
1	I1	Interrupt associated with TCC #1	W	0x0
0	I0	Interrupt associated with TCC #0	W	0x0

Table 16-318. Register Call Summary for Register EDMA_TPCC_ICR_RN_k

Enhanced DMA

- EDMA Register Summary: [0] [1] [2] [3] [4] [5]

Table 16-319. EDMA_TPCC_ICRH_RN_k

Address Offset	0x0000 2074 + (0x200 * k)		
Physical Address	0x4330 2074 + (0x200 * k) 0x40D1 2074 + (0x200 * k) 0x4151 2074 + (0x200 * k) 0x01D1 2074 + (0x200 * k) 0x420A 2074 + (0x200 * k) 0x421A 2074 + (0x200 * k) 0x422A 2074 + (0x200 * k) 0x423A 2074 + (0x200 * k)	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Interrupt Clear Register (High Part) CPU write of '0' has no effect. CPU write of '1' to the EDMA_TPCC_ICRH.In bit causes the EDMA_TPCC_IPRH.In bit to be cleared. All EDMA_TPCC_IPRH.In bits must be cleared before additional interrupts will be asserted by CC.		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
I63	I62	I61	I60	I59	I58	I57	I56	I55	I54	I53	I52	I51	I50	I49	I48	I47	I46	I45	I44	I43	I42	I41	I40	I39	I38	I37	I36	I35	I34	I33	I32

Bits	Field Name	Description	Type	Reset
31	I63	Interrupt associated with TCC #63	W	0x0
30	I62	Interrupt associated with TCC #62	W	0x0
29	I61	Interrupt associated with TCC #61	W	0x0
28	I60	Interrupt associated with TCC #60	W	0x0
27	I59	Interrupt associated with TCC #59	W	0x0
26	I58	Interrupt associated with TCC #58	W	0x0
25	I57	Interrupt associated with TCC #57	W	0x0
24	I56	Interrupt associated with TCC #56	W	0x0
23	I55	Interrupt associated with TCC #55	W	0x0
22	I54	Interrupt associated with TCC #54	W	0x0
21	I53	Interrupt associated with TCC #53	W	0x0
20	I52	Interrupt associated with TCC #52	W	0x0
19	I51	Interrupt associated with TCC #51	W	0x0
18	I50	Interrupt associated with TCC #50	W	0x0
17	I49	Interrupt associated with TCC #49	W	0x0
16	I48	Interrupt associated with TCC #48	W	0x0
15	I47	Interrupt associated with TCC #47	W	0x0
14	I46	Interrupt associated with TCC #46	W	0x0
13	I45	Interrupt associated with TCC #45	W	0x0

Bits	Field Name	Description	Type	Reset
12	I44	Interrupt associated with TCC #44	W	0x0
11	I43	Interrupt associated with TCC #43	W	0x0
10	I42	Interrupt associated with TCC #42	W	0x0
9	I41	Interrupt associated with TCC #41	W	0x0
8	I40	Interrupt associated with TCC #40	W	0x0
7	I39	Interrupt associated with TCC #39	W	0x0
6	I38	Interrupt associated with TCC #38	W	0x0
5	I37	Interrupt associated with TCC #37	W	0x0
4	I36	Interrupt associated with TCC #36	W	0x0
3	I35	Interrupt associated with TCC #35	W	0x0
2	I34	Interrupt associated with TCC #34	W	0x0
1	I33	Interrupt associated with TCC #33	W	0x0
0	I32	Interrupt associated with TCC #32	W	0x0

Table 16-320. Register Call Summary for Register EDMA_TPCC_ICRH_RN_k

Enhanced DMA

- EDMA Register Summary: [0] [1] [2] [3] [4] [5]

Table 16-321. EDMA_TPCC_IEVAL_RN_k

Address Offset	0x0000 2078 + (0x200 * k)		
Physical Address	0x4330 2078 + (0x200 * k) 0x40D1 2078 + (0x200 * k) 0x4151 2078 + (0x200 * k) 0x01D1 2078 + (0x200 * k) 0x420A 2078 + (0x200 * k) 0x421A 2078 + (0x200 * k) 0x422A 2078 + (0x200 * k) 0x423A 2078 + (0x200 * k)	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Interrupt Eval Register		
Type	W		

31:2	1	0
RESERVED	SET	EVAL

Bits	Field Name	Description	Type	Reset
31:2	RESERVED	Reserved	R	0x0
1	SET	Interrupt Set CPU writes 0x0 has no effect. CPU writes 0x1 to the SETn bit causes the tpcc_intN output signal to be pulsed egardless of state of interrupts enable (IERn) and status (EDMA_TPCC_IPRn).	W	0x0
0	EVAL	Interrupt Evaluate CPU writes 0x0 has no effect. CPU writes 0x1 to the EVALn bit causes the tpcc_intN output signal to be pulsed if any enabled interrupts (IERn) are still pending (EDMA_TPCC_IPRn).	W	0x0

Table 16-322. Register Call Summary for Register EDMA_TPCC_IEVAL_RN_k

Enhanced DMA
- Region Overview: [0] [1]
- EDMA Register Summary: [2] [3] [4] [5] [6] [7]

Table 16-323. EDMA_TPCC_QER_RN_k

Address Offset	0x0000 2080 + (0x200 * k)		
Physical Address	0x4330 2080 + (0x200 * k) 0x40D1 2080 + (0x200 * k) 0x4151 2080 + (0x200 * k) 0x01D1 2080 + (0x200 * k) 0x420A 2080 + (0x200 * k) 0x421A 2080 + (0x200 * k) 0x422A 2080 + (0x200 * k) 0x423A 2080 + (0x200 * k)	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	QDMA Event Register If EDMA_TPCC_QER.En bit is set, then the corresponding QDMA channel is prioritized vs. other qdma events for submission to the TC. EDMA_TPCC_QER.En bit is set when a vbus write byte matches the address defined in the QCHMAPn register. EDMA_TPCC_QER.En bit is cleared when the corresponding event is prioritized and serviced. EDMA_TPCC_QER.En is also cleared when user writes a '1' to the EDMA_TPCC_QSECR.En bit. If the EDMA_TPCC_QER.En bit is already set and a new QDMA event is detected due to user write to QDMA trigger location and EDMA_TPCC_QEER register is set, then the corresponding bit in the QDMA Event Missed Register is set.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	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

Bits	Field Name	Description	Type	Reset
31:8	RESERVED	Reserved	R Return 0's	0x0
7	E7	Event #7	R	0x0
6	E6	Event #6	R	0x0
5	E5	Event #5	R	0x0
4	E4	Event #4	R	0x0
3	E3	Event #3	R	0x0
2	E2	Event #2	R	0x0
1	E1	Event #1	R	0x0
0	E0	Event #0	R	0x0

Table 16-324. Register Call Summary for Register EDMA_TPCC_QER_RN_k

Enhanced DMA
- EDMA Register Summary: [0] [1] [2] [3] [4] [5]

Table 16-325. EDMA_TPCC_QEER_RN_k

Address Offset	0x0000 2084 + (0x200 * k)		
Physical Address	0x4330 2084 + (0x200 * k) 0x40D1 2084 + (0x200 * k) 0x4151 2084 + (0x200 * k) 0x01D1 2084 + (0x200 * k) 0x420A 2084 + (0x200 * k) 0x421A 2084 + (0x200 * k) 0x422A 2084 + (0x200 * k) 0x423A 2084 + (0x200 * k)	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	QDMA Event Enable Register Enabled/disabled QDMA address comparator for QDMA Channel N. EDMA_TPCC_QEER.En is not directly writeable. The corresponding QDMA channel comparator is enabled and Events will be recognized and latched in EDMA_TPCC_QER.En. EDMA_TPCC_QEER.En = 0, The corresponding QDMA channel comparator is disabled. Events will not be recognized/latched in EDMA_TPCC_QER.En. QDMA channels can be enabled via writes to EDMA_TPCC_QEESR and can be disabled via writes to EDMA_TPCC_QEECR register. EDMA_TPCC_QEER.En = 1,		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	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

Bits	Field Name	Description	Type	Reset
31:8	RESERVED	Reserved	R Return 0's	0x0
7	E7	Event #7	R	0x0
6	E6	Event #6	R	0x0
5	E5	Event #5	R	0x0
4	E4	Event #4	R	0x0
3	E3	Event #3	R	0x0
2	E2	Event #2	R	0x0
1	E1	Event #1	R	0x0
0	E0	Event #0	R	0x0

Table 16-326. Register Call Summary for Register EDMA_TPCC_QEER_RN_k

Enhanced DMA

- EDMA Register Summary: [0] [1] [2] [3] [4] [5]

Table 16-327. EDMA_TPCC_QEECR_RN_k

Address Offset	0x0000 2088 + (0x200 * k)		
Physical Address	0x4330 2088 + (0x200 * k) 0x40D1 2088 + (0x200 * k) 0x4151 2088 + (0x200 * k) 0x01D1 2088 + (0x200 * k) 0x420A 2088 + (0x200 * k) 0x421A 2088 + (0x200 * k) 0x422A 2088 + (0x200 * k) 0x423A 2088 + (0x200 * k)	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	QDMA Event Enable Clear Register CPU write of '0' has no effect. CPU write of '1' to the EDMA_TPCC_QEECR.En bit causes the EDMA_TPCC_QEER.En bit to be cleared.		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	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

Bits	Field Name	Description	Type	Reset
31:8	RESERVED	Reserved	R	0x0
7	E7	Event #7	W	0x0
6	E6	Event #6	W	0x0
5	E5	Event #5	W	0x0
4	E4	Event #4	W	0x0
3	E3	Event #3	W	0x0
2	E2	Event #2	W	0x0
1	E1	Event #1	W	0x0
0	E0	Event #0	W	0x0

Table 16-328. Register Call Summary for Register EDMA_TPCC_QEECR_RN_k

Enhanced DMA

- EDMA Register Summary: [0] [1] [2] [3] [4] [5]

Table 16-329. EDMA_TPCC_QEESR_RN_k

Address Offset	0x0000 208C + (0x200 * k)		
Physical Address	0x4330 208C + (0x200 * k) 0x40D1 208C + (0x200 * k) 0x4151 208C + (0x200 * k) 0x01D1 208C + (0x200 * k) 0x420A 208C + (0x200 * k) 0x421A 208C + (0x200 * k) 0x422A 208C + (0x200 * k) 0x423A 208C + (0x200 * k)	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	QDMA Event Enable Set Register CPU write of '0' has no effect. CPU write of '1' to the EDMA_TPCC_QEESR.En bit causes the EDMA_TPCC_QEESR.En bit to be set.		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	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

Bits	Field Name	Description	Type	Reset
31:8	RESERVED	Reserved	R	0x0
7	E7	Event #7	W	0x0
6	E6	Event #6	W	0x0
5	E5	Event #5	W	0x0
4	E4	Event #4	W	0x0
3	E3	Event #3	W	0x0
2	E2	Event #2	W	0x0
1	E1	Event #1	W	0x0
0	E0	Event #0	W	0x0

Table 16-330. Register Call Summary for Register EDMA_TPCC_QEESR_RN_k

Enhanced DMA

- EDMA Register Summary: [0] [1] [2] [3] [4] [5]

Table 16-331. EDMA_TPCC_QSER_RN_k

Address Offset	0x0000 2090 + (0x200 * k)		
Physical Address	0x4330 2090 + (0x200 * k) 0x40D1 2090 + (0x200 * k) 0x4151 2090 + (0x200 * k) 0x01D1 2090 + (0x200 * k) 0x420A 2090 + (0x200 * k) 0x421A 2090 + (0x200 * k) 0x422A 2090 + (0x200 * k) 0x423A 2090 + (0x200 * k)	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	QDMA Secondary Event Register The QDMA secondary event register is used along with the QDMA Event Register (EDMA_TPCC_QER) to provide information on the state of a QDMA Event. En = 0 : Event is not currently in the Event Queue. En = 1 : Event is currently stored in Event Queue. Event arbiter will not prioritize additional events.		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	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

Bits	Field Name	Description	Type	Reset
31:8	RESERVED	Reserved	R Return 0's	0x0
7	E7	Event #7	R	0x0
6	E6	Event #6	R	0x0
5	E5	Event #5	R	0x0
4	E4	Event #4	R	0x0
3	E3	Event #3	R	0x0
2	E2	Event #2	R	0x0
1	E1	Event #1	R	0x0
0	E0	Event #0	R	0x0

Table 16-332. Register Call Summary for Register EDMA_TPCC_QSER_RN_k

Enhanced DMA

- EDMA Register Summary: [0] [1] [2] [3] [4] [5]

Table 16-333. EDMA_TPCC_QSECR_RN_k

Address Offset	0x0000 2094 + (0x200 * k)		
Physical Address	0x4330 2094 + (0x200 * k) 0x40D1 2094 + (0x200 * k) 0x4151 2094 + (0x200 * k) 0x01D1 2094 + (0x200 * k) 0x420A 2094 + (0x200 * k) 0x421A 2094 + (0x200 * k) 0x422A 2094 + (0x200 * k) 0x423A 2094 + (0x200 * k)	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	QDMA Secondary Event Clear Register CPU write of '0' has no effect. The secondary event clear register is used to clear the status of the EDMA_TPCC_QSER and EDMA_TPCC_QER register (note that this is slightly different than the EDMA_TPCC_SER operation, which does not clear the EDMA_TPCC_ER.En register). CPU write of '1' to the EDMA_TPCC_QSECR.En bit clears the EDMA_TPCC_QSER.En and EDMA_TPCC_QER.En register fields.		
Type	W		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	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

Bits	Field Name	Description	Type	Reset
31:8	RESERVED	write 0's for future compatibility	W	0x0
7	E7	Event #7	W	0x0
6	E6	Event #6	W	0x0
5	E5	Event #5	W	0x0
4	E4	Event #4	W	0x0
3	E3	Event #3	W	0x0
2	E2	Event #2	W	0x0
1	E1	Event #1	W	0x0
0	E0	Event #0	W	0x0

Table 16-334. Register Call Summary for Register EDMA_TPCC_QSECR_RN_k

Enhanced DMA

- EDMA Register Summary: [0] [1] [2] [3] [4] [5]

Table 16-335. EDMA_TPCC_OPT_n

Address Offset	0x0000 4000 + (0x20 * n)		
Physical Address	0x4330 4000 + (0x20 * n) 0x40D1 4000 + (0x20 * n) 0x4151 4000 + (0x20 * n) 0x01D1 4000 + (0x20 * n) 0x420A 4000 + (0x20 * n) 0x421A 4000 + (0x20 * n) 0x422A 4000 + (0x20 * n) 0x423A 4000 + (0x20 * n)	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Options Parameter		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRIV	RESERVED			PRIVID				ITCCHEN	TCCHEN	ITCINTEN	TCINTEN	WIMODE	RESERVED	TCC						TCCMODE	FWID			RESERVED				STATIC	SYNCDIM	DAM	SAM

Bits	Field Name	Description	Type	Reset
31	PRIV	Privilege level privilege level (supervisor vs. user) for the host/cpu/dma that programmed this PaRAM Entry. Value is set with the vbus priv value when any part of the PaRAM Entry is written. Not writeable via vbus wdata bus. Is readable via VBus rdata bus. 0x0: User level privilege 0x1: Supervisor level privilege	R	0x0
30:28	RESERVED	Reserved	R	0x0
27:24	PRIVID	Privilege ID Privilege ID for the external host/cpu/dma that programmed this PaRAM Entry. This value is set with the vbus privid value when any part of the PaRAM Entry is written. Not writeable via vbus wdata bus. Is readable via VBus rdata bus.	R	0x0

Bits	Field Name	Description	Type	Reset
23	ITCCHEN	Intermediate transfer completion chaining enable 0x0: Intermediate transfer complete chaining is disabled. 0x1: Intermediate transfer complete chaining is enabled.	RW	0x0
22	TCCHEN	Transfer complete chaining enable 0x0: Transfer complete chaining is disabled. 0x1: Transfer complete chaining is enabled.	RW	0x0
21	ITCINTEN	Intermediate transfer completion interrupt enable 0x0: Intermediate transfer complete interrupt is disabled. 0x1: Intermediate transfer complete interrupt is enabled (corresponding EDMA_TPCC_IER[TCC] bit must be set to 1 to generate interrupt)	RW	0x0
20	TCINTEN	Transfer complete interrupt enable 0x0: Transfer complete interrupt is disabled. 0x1: Transfer complete interrupt is enabled (corresponding EDMA_TPCC_IER[TCC] bit must be set to 1 to generate interrupt)	RW	0x0
19	WIMODE	Backward compatibility mode 0x0: Normal operation 0x1: WI Backwards Compatibility mode, forces BCNT to be adjusted by '1' upon TR submission (0 means 1, 1 means 2, ...) and forces ACNT to be treated as a word-count (left shifted by 2 by hardware to create byte cnt for TR submission)	RW	0x0
18	RESERVED	Reserved	R	0x0
17:12	TCC	Transfer Complete Code The 6-bit code is used to set the relevant bit in EDMA_TPCC_CER (bit EDMA_TPCC_CER[TCC]) for chaining or in EDMA_TPCC_IER (bit EDMA_TPCC_IER[TCC]) for interrupts.	RW	0x0
11	TCCMODE	Transfer complete code mode: Indicates the point at which a transfer is considered completed. Applies to both chaining and interrupt. 0x0: Normal Completion. A transfer is considered completed after the transfer parameters are returned to the CC from the TC (which was returned from the peripheral) 0x1: Early Completion, A transfer is considered completed after the CC submits a TR to the TC. CC generates completion code internally.	RW	0x0
10:8	FWID	FIFO width: Applies if either SAM or DAM is set to FIFO mode. Pass-thru to TC.	RW	0x0
7:4	RESERVED	Reserved	R	0x0
3	STATIC	Static Entry 0x0: Entry is updated as normal 0x1: Entry is static, Count and Address updates are not updated after TRP is submitted. Linking is not performed.	RW	0x0
2	SYNCDIM	Transfer Synchronization Dimension: 0x0: A-Sync, Each event triggers the transfer of ACNT elements. 0x1: AB-Sync, Each event triggers the transfer of BCNT arrays of ACNT elements.	RW	0x0
1	DAM	Destination Address Mode: Destination Address Mode within an array. Pass-thru to TC. 0x0: INCR, Dst addressing within an array increments. Dst is not a FIFO. 0x1: FIFO, Dst addressing within an array wraps around upon reaching FIFO width.	RW	0x0

Bits	Field Name	Description	Type	Reset
0	SAM	Source Address Mode: Source Address Mode within an array. Pass-thru to TC. 0x0: INCR, Src addressing within an array increments. Source is not a FIFO. 0x1: FIFO, Src addressing within an array wraps around upon reaching FIFO width.	RW	0x0

Table 16-336. Register Call Summary for Register EDMA_TPCC_OPT_n

Enhanced DMA

- Types of EDMA controller Transfers: [0]
- PaRAM: [1]
- EDMA Channel PaRAM Set Entry Fields: [2] [3] [4] [5]
- Dummy Versus Null Transfer Comparison: [6]
- Parameter Set Updates: [7] [8] [9] [10] [11] [12]
- Linking Transfers: [13] [14] [15]
- Constant Addressing Mode Transfers/Alignment Issues: [16] [17] [18] [19] [20] [21]
- DMA Channel: [22]
- Comparison Between DMA and QDMA Channels: [23]
- Completion of a DMA Transfer: [24] [25] [26] [27] [28] [29] [30]
- Normal Completion: [31]
- Early Completion: [32]
- Channel Controller Regions: [33]
- Chaining EDMA Channels: [34] [35] [36] [37] [38] [39] [40] [41] [42] [43] [44] [45]
- Transfer Completion Interrupts: [46] [47] [48] [49] [50] [51] [52] [53] [54] [55] [56] [57] [58] [59] [60] [61] [62] [63] [64] [65] [66] [67] [68] [69] [70] [71]
- Error Interrupts: [72] [73]
- Active Memory Protection: [74] [75] [76] [77]
- Proxy Memory Protection: [78] [79] [80] [81] [82] [83] [84] [85] [86] [87]
- Block Move Example: [88] [89] [90]
- Subframe Extraction Example: [91] [92] [93] [94]
- Data Sorting Example: [95] [96] [97] [98]
- Non-bursting Peripherals: [99]
- Bursting Peripherals: [100] [101]
- Continuous Operation: [102] [103] [104] [105]
- Ping-Pong Buffering: [106] [107] [108]
- Transfer Chaining Examples: [109] [110] [111] [112] [113] [114] [115] [116] [117]
- EDMA Debug Checklist: [118] [119] [120] [121] [122] [123]
- EDMA Register Summary: [124] [125] [126] [127] [128] [129]
- EDMA Register Description: [130] [131] [132] [133] [134] [135] [136] [137] [138]

Table 16-337. EDMA_TPCC_SRC_n

Address Offset	0x0000 4004 + (0x20 * n)		
Physical Address	0x4330 4004 + (0x20 * n) 0x40D1 4004 + (0x20 * n) 0x4151 4004 + (0x20 * n) 0x01D1 4004 + (0x20 * n) 0x420A 4004 + (0x20 * n) 0x421A 4004 + (0x20 * n) 0x422A 4004 + (0x20 * n) 0x423A 4004 + (0x20 * n)	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Source Address		
Type	RW		

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

Bits	Field Name	Description	Type	Reset
31:0	SRC	Source Address The 32-bit source address parameters specify the starting byte address of the source. If SAM is set to FIFO mode then the user should program the Source address to be aligned to the value specified by the EDMA_TPCC_OPT_n[10:8] FWID field. No errors are recognized here but TC will assert error if this is not true.	RW	0x0

Table 16-338. Register Call Summary for Register EDMA_TPCC_SRC_n

Enhanced DMA
- PaRAM: [0]
- Parameter Set Updates: [1] [2]
- EDMA Register Summary: [3] [4] [5] [6] [7] [8]

Table 16-339. EDMA_TPCC_ABCNT_n

Address Offset	0x0000 4008 + (0x20 * n)		
Physical Address	0x4330 4008 + (0x20 * n) 0x40D1 4008 + (0x20 * n) 0x4151 4008 + (0x20 * n) 0x01D1 4008 + (0x20 * n) 0x420A 4008 + (0x20 * n) 0x421A 4008 + (0x20 * n) 0x422A 4008 + (0x20 * n) 0x423A 4008 + (0x20 * n)	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	A and B byte count		
Type	RW		

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

Bits	Field Name	Description	Type	Reset
31:16	BCNT	BCNT: Count for 2nd Dimension: BCNT is a 16-bit unsigned value that specifies the number of arrays of length ACNT. For normal operation, valid values for BCNT can be anywhere between 1 and 65535. Therefore, the maximum number of arrays in a frame is 65535 (64K-1 arrays). BCNT=1 means 1 array in the frame, and BCNT=0 means 0 arrays in the frame. In normal mode, a BCNT of '0' is considered as either a Null or Dummy transfer. A Dummy or Null transfer will generate a Completion code depending on the settings of the completion bit fields of the OPT field. If the EDMA_TPCC_OPT_n.WIMODE bit is set, then the programmed BCNT value will be incremented by '1' before submission to TC. I.e., 0 means 1, 1 means 2, 2 means 3, ..., 0xFFFE means 0xFFFF. A value of 0xFFFF is an illegal value that will be treated as a Null TR.	RW	0x0

Bits	Field Name	Description	Type	Reset
15:0	ACNT	ACNT: number of bytes in 1st dimension: ACNT represents the number of bytes within the first dimension of a transfer. ACNT is a 16-bit unsigned value with valid values between 0 and 65535. Therefore, the maximum number of bytes in an array is 65535 bytes (64K-1 bytes). ACNT must be greater than or equal to '1' for a TR to be submitted to TC. An ACNT of '0' is considered as either a null or dummy transfer. A Dummy or Null transfer will generate a Completion code depending on the settings of the completion bit fields of the OPT field. If the EDMA_TPCC_OPT_n.WIMODE bit is set then the ACNT field represents a word count. The CC must internally multiply by 4 to translate the word count to a byte count prior to submission to the TC. The 2 MSBs of the 16-bit ACNT are reserved and should always be written as 'b00 by the user. If user writes a value other than 0, it will still be treated as 0 since the multiply-by-4 operation (to translate between a word count and a byte count) will drop the 2 msbits. For dummy and null transfer definition, the ACNT definition will disregard the 2 msbits. I.e., a programmed ACNT value of 0x8000 in WI-mode will be treated as 0 byte transfer, resulting in null or dummy operation dependent on the state of BCNT and CCNT.	RW	0x0

Table 16-340. Register Call Summary for Register EDMA_TPCC_ABCNT_n

Enhanced DMA

- Types of EDMA controller Transfers: [0] [1]
- A-Synchronized Transfers: [2]
- PaRAM: [3] [4]
- EDMA Channel PaRAM Set Entry Fields: [5] [6] [7]
- Null PaRAM Set: [8] [9]
- Dummy PaRAM Set: [10] [11]
- Parameter Set Updates: [12] [13] [14] [15] [16] [17] [18] [19] [20] [21] [22] [23] [24]
- Element Size: [25] [26] [27] [28] [29] [30]
- Comparison Between DMA and QDMA Channels: [31]
- Completion of a DMA Transfer: [32] [33]
- Event, Channel, and PaRAM Mapping: [34] [35]
- Chaining EDMA Channels: [36] [37]
- Transfer Completion Interrupts: [38]
- Architecture Details: [39] [40] [41] [42] [43] [44] [45]
- Block Move Example: [46]
- Bursting Peripherals: [47] [48]
- EDMA Register Summary: [49] [50] [51] [52] [53] [54]

Table 16-341. EDMA_TPCC_DST_n

Address Offset	0x0000 400C + (0x20 * n)		
Physical Address	0x4330 400C + (0x20 * n) 0x40D1 400C + (0x20 * n) 0x4151 400C + (0x20 * n) 0x01D1 400C + (0x20 * n) 0x420A 400C + (0x20 * n) 0x421A 400C + (0x20 * n) 0x422A 400C + (0x20 * n) 0x423A 400C + (0x20 * n)	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Destination Address		
Type	RW		

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

Bits	Field Name	Description	Type	Reset
31:0	DST	Destination Address: The 32-bit destination address parameters specify the starting byte address of the destination. If DAM is set to FIFO mode then the user should program the Destination address to be aligned to the value specified by the EDMA_TPCC_OPT_n.FWID field. No errors are recognized here but TC will assert error if this is not true.	RW	0x0

Table 16-342. Register Call Summary for Register EDMA_TPCC_DST_n

Enhanced DMA

- PaRAM: [0]
- Parameter Set Updates: [1] [2]
- EDMA Register Summary: [3] [4] [5] [6] [7] [8]

Table 16-343. EDMA_TPCC_BIDX_n

Address Offset	0x0000 4010 + (0x20 * n)		
Physical Address	0x4330 4010 + (0x20 * n) 0x40D1 4010 + (0x20 * n) 0x4151 4010 + (0x20 * n) 0x01D1 4010 + (0x20 * n) 0x420A 4010 + (0x20 * n) 0x421A 4010 + (0x20 * n) 0x422A 4010 + (0x20 * n) 0x423A 4010 + (0x20 * n)	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description			
Type	RW		

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

Bits	Field Name	Description	Type	Reset
31:16	DBIDX	Destination 2nd Dimension Index: DBIDX is a 16-bit signed value (2's complement) used for destination address modification in between each array in the 2nd dimension. It is a signed value between -32768 and 32767. 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-Sync and AB-Sync transfers.	RW	0x0
15:0	SBIDX	Source 2nd Dimension Index: SBIDX is a 16-bit signed value (2's complement) used for source address modification in between each array in the 2nd dimension. It is a signed value between -32768 and 32767. 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-sync and AB-sync transfers.	RW	0x0

Table 16-344. Register Call Summary for Register EDMA_TPCC_BIDX_n

Enhanced DMA
- Types of EDMA controller Transfers: [0] [1]
- A-Synchronized Transfers: [2] [3] [4]
- AB-Synchronized Transfers: [5] [6]
- PaRAM: [7] [8]
- EDMA Channel PaRAM Set Entry Fields: [9] [10] [11] [12] [13] [14] [15]
- Parameter Set Updates: [16] [17] [18] [19] [20] [21] [22] [23] [24] [25] [26] [27] [28] [29]
- Constant Addressing Mode Transfers/Alignment Issues: [30]
- Architecture Details: [31]
- Bursting Peripherals: [32]
- EDMA Register Summary: [33] [34] [35] [36] [37] [38]

Table 16-345. EDMA_TPCC_LNK_n

Address Offset	0x0000 4014 + (0x20 * n)		
Physical Address	0x4330 4014 + (0x20 * n) 0x40D1 4014 + (0x20 * n) 0x4151 4014 + (0x20 * n) 0x01D1 4014 + (0x20 * n) 0x420A 4014 + (0x20 * n) 0x421A 4014 + (0x20 * n) 0x422A 4014 + (0x20 * n) 0x423A 4014 + (0x20 * n)	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Link and Reload parameters		
Type	RW		

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

Bits	Field Name	Description	Type	Reset
31:16	BCNTRLD	BCNT Reload: 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-Sync'ed transfers. In this case, the CC decrements the BCNT value by one on each TR submission. When BCNT (conceptually) reaches zero, then the CC decrements CCNT and uses the BCNTRLD value to reinitialize the BCNT value. For AB-synchronized transfers, the CC submits the BCNT in the TR and therefore the TC is responsible to keep track of BCNT, not thus BCNTRLD is a don't care field.	RW	0x0

Bits	Field Name	Description	Type	Reset
15:0	LINK	Link Address: The CC provides a mechanism to reload the current PaRAM Entry upon its natural termination (i.e., after count fields are decremented to '0') with a new PaRAM Entry. This is called 'linking'. The 16-bit parameter LINK specifies the byte address offset in the PaRAM from which the CC loads/reloads the next PaRAM entry in the link. The CC should disregard the value in the upper 2 bits of the LINK field as well as the lower 5-bits of the LINK field. The upper two bits are ignored such that the user can program either the 'literal' byte address of the LINK parameter or the 'PaRAM base-relative' address of the link field. Therefore, if the user uses the literal address with a range from 0x4000 to 0x7FFF, it will be treated as a PaRAM-base-relative value of 0x0000 to 0x3FFF. The lower-5 bits are ignored and treated as 'b00000, thereby guaranteeing that all Link pointers point to a 32-byte aligned PaRAM entry. In the latter case (5-lsbs), behavior is undefined for the user (i.e., don't have to test it). In the former case (2 msbs), user should be able to take advantage of this feature (i.e., do have to test it). If a Link Update is requested to a PaRAM address that is beyond the actual range of implemented PaRAM, then the Link will be treated as a Null Linkand all 0s plus 0xFFFF will be written to the current entry location. A LINK value of 0xFFFF is referred to as a NULL link which should cause the CC to write 0x0 to all entries of the current PaRAM Entry except for the LINK field which is set to 0xFFFF. The Priv/Privid state is overwritten to 0x0 when linking. MSBs and LSBS should not be masked when comparing against the 0xFFFF value. I.e., a value of 0x3FFE is a non-NULL PaRAM link field.	RW	0x0

Table 16-346. Register Call Summary for Register EDMA_TPCC_LNK_n

Enhanced DMA

- PaRAM: [0] [1]
- EDMA Channel PaRAM Set Entry Fields: [2] [3] [4] [5] [6]
- Parameter Set Updates: [7] [8] [9] [10] [11] [12] [13] [14]
- Linking Transfers: [15] [16] [17] [18] [19] [20]
- EDMA Debug Checklist: [21]
- EDMA Register Summary: [22] [23] [24] [25] [26] [27]

Table 16-347. EDMA_TPCC_CIDX_n

Address Offset	0x0000 4018 + (0x20 * n)		
Physical Address	0x4330 4018 + (0x20 * n) 0x40D1 4018 + (0x20 * n) 0x4151 4018 + (0x20 * n) 0x01D1 4018 + (0x20 * n) 0x420A 4018 + (0x20 * n) 0x421A 4018 + (0x20 * n) 0x422A 4018 + (0x20 * n) 0x423A 4018 + (0x20 * n)	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	Source and destination frame indexes		
Type	RW		

31–16	15–0
DCIDX	SCIDX

Bits	Field Name	Description	Type	Reset
31:16	DCIDX	Destination Frame Index: DCIDX is a 16-bit signed value (2's complement) used for destination address modification for the 3rd dimension. It is a signed value between -32768 and 32767. 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 in the next frame. It applies to both A-sync and AB-sync transfers. Note that when DCIDX is applied, the current array in an A-sync transfer is the last array in the frame, while the current array in a ABsync transfer is the first array in the frame.	RW	0x0
15:0	SCIDX	Source Frame Index: SCIDX is a 16-bit signed value (2's complement) used for source address modification for the 3rd dimension. It is a signed value between -32768 and 32767. 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-sync and AB-sync transfers. Note that when SCIDX is applied, the current array in an A-sync transfer is the last array in the frame, while the current array in a AB-sync transfer is the first array in the frame.	RW	0x0

Table 16-348. Register Call Summary for Register EDMA_TPCC_CIDX_n

Enhanced DMA

- Types of EDMA controller Transfers: [0] [1]
- A-Synchronized Transfers: [2] [3]
- AB-Synchronized Transfers: [4] [5]
- PaRAM: [6] [7]
- EDMA Channel PaRAM Set Entry Fields: [8] [9] [10]
- Parameter Set Updates: [11] [12] [13] [14]
- Bursting Peripherals: [15] [16] [17]
- EDMA Register Summary: [18] [19] [20] [21] [22] [23]

Table 16-349. EDMA_TPCC_CCNT_n

Address Offset	0x0000 401C + (0x20 * n)		
Physical Address	0x4330 401C + (0x20 * n) 0x40D1 401C + (0x20 * n) 0x4151 401C + (0x20 * n) 0x01D1 401C + (0x20 * n) 0x420A 401C + (0x20 * n) 0x421A 401C + (0x20 * n) 0x422A 401C + (0x20 * n) 0x423A 401C + (0x20 * n)	**Instance**	SYS_EDMA_TPCC DSP1_EDMA_TPCC DSP2_EDMA_TPCC DSP_EDMA_TPCC EVE1_EDMA_TPCC EVE2_EDMA_TPCC EVE3_EDMA_TPCC EVE4_EDMA_TPCC
Description	C byte count		
Type	RW		

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

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Reserved	RW	0x0
15:0	CCNT	CCNT: Count for 3rd Dimension: CCNT is a 16-bit unsigned value that specifies the number of frames in a block. Valid values for CCNT can be anywhere between 1 and 65535. Therefore, the maximum number of frames in a block is 65535 (64K-1 frames). CCNT of '1' means '1' frame in the block, and CCNT of '0' means '0' frames in the block. A CCNT value of '0' is considered as either a null or dummy transfer. A Dummy or Null transfer will generate a Completion code depending on the settings of the completion bit fields of the OPT field. WIMODE has no affect on CCNT operation.	RW	0x0

Table 16-350. Register Call Summary for Register EDMA_TPCC_CCNT_n

Enhanced DMA

- Types of EDMA controller Transfers: [0]
- AB-Synchronized Transfers: [1]
- PaRAM: [2]
- EDMA Channel PaRAM Set Entry Fields: [3]
- Null PaRAM Set: [4]
- Dummy PaRAM Set: [5]
- Parameter Set Updates: [6] [7] [8] [9] [10] [11] [12]
- Element Size: [13]
- Comparison Between DMA and QDMA Channels: [14] [15]
- Completion of a DMA Transfer: [16] [17] [18]
- Event, Channel, and PaRAM Mapping: [19]
- Chaining EDMA Channels: [20]
- Transfer Completion Interrupts: [21]
- Bursting Peripherals: [22]
- EDMA Programming Tips: [23]
- EDMA Register Summary: [24] [25] [26] [27] [28] [29]

16.2.8.2.2.2 EDMA_TPTC0 and EDMA_TPTC1 Register Description

Table 16-351 through Table 16-423 describe the individual EDMA_TPTC0 and EDMA_TPTC1 module registers.

Table 16-351. EDMA_TPTCn_PID

Address Offset	0x0000 0000		
Physical Address	0x4340 0000 0x4350 0000 0x40D0 5000 0x40D0 6000 0x4150 5000 0x4150 6000 0x01D0 5000 0x01D0 6000	**Instance**	SYS_EDMA_TPTC0 SYS_EDMA_TPTC1 DSP1_EDMA_TPTC0 DSP1_EDMA_TPTC1 DSP2_EDMA_TPTC0 DSP2_EDMA_TPTC1 DSP_EDMA_TPTC0 DSP_EDMA_TPTC1
Description	Peripheral ID Register		
Type	R		

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

Bits	Field Name	Description	Type	Reset
31:0	REVISION	IP Revision	R	TI internal data

Table 16-352. Register Call Summary for Register EDMA_TPTCn_PID

Enhanced DMA
- EDMA Register Summary: [0] [1] [2] [3] [4] [5] [6] [7]

Table 16-353. EDMA_TPTCn_TCCFG

Address Offset	0x0000 0004		
Physical Address	0x4340 0004 0x4350 0004 0x40D0 5004 0x40D0 6004 0x4150 5004 0x4150 6004 0x01D0 5004 0x01D0 6004	**Instance**	SYS_EDMA_TPTC0 SYS_EDMA_TPTC1 DSP1_EDMA_TPTC0 DSP1_EDMA_TPTC1 DSP2_EDMA_TPTC0 DSP2_EDMA_TPTC1 DSP_EDMA_TPTC0 DSP_EDMA_TPTC1
Description	TC Configuration Register		
Type	R		

31:10	9:8	7:6	5:4	3	2:0
RESERVED	DREGDEPTH	RESERVED	BUSWIDTH	RESERVED	FIFOSIZE

Bits	Field Name	Description	Type	Reset
31:10	RESERVED	Reads return 0's	R	0x0
9:8	DREGDEPTH	Dst Register FIFO Depth Parameterization 0x0: 1 entry 0x1: 2 entries 0x2: 4 entries	R	See Table 16-84
7:6	RESERVED	Reads return 0's	R	0x0
5:4	BUSWIDTH	Bus Width Parameterization 0x0: 32-bit 0x1: 64-bit 0x2: 128-bit	R	See Table 16-84
3	RESERVED	Reads return 0's	R	0x0
2:0	FIFOSIZE	Fifo Size Parameterization 0x0: 32 byte FIFO 0x1: 64 byte FIFO 0x2: 128 byte FIFO 0x3: 256 byte FIFO 0x4: 512 byte FIFO	R	See Table 16-84

Table 16-354. Register Call Summary for Register EDMA_TPTCn_TCCFG

Enhanced DMA
- EDMA_TPTC Configuration: [1] [2] [3]
- EDMA Register Summary: [4] [5] [6] [7] [8] [9] [10] [11]

Table 16-355. EDMA_TPTCn_TCSTAT

Address Offset	0x0000 0100		
Physical Address	0x4340 0100 0x4350 0100 0x40D0 5100 0x40D0 6100 0x4150 5100 0x4150 6100 0x01D0 5100 0x01D0 6100 0x4208 6100 0x4218 6100 0x4208 7100 0x4218 7100 0x4228 6100 0x4238 6100 0x4228 7100 0x4238 7100	**Instance**	SYS_EDMA_TPTC0 SYS_EDMA_TPTC1 DSP1_EDMA_TPTC0 DSP1_EDMA_TPTC1 DSP2_EDMA_TPTC0 DSP2_EDMA_TPTC1 DSP_EDMA_TPTC0 DSP_EDMA_TPTC1 EVE1_EDMA_TPTC0 EVE2_EDMA_TPTC0 EVE1_EDMA_TPTC1 EVE2_EDMA_TPTC1 EVE3_EDMA_TPTC0 EVE4_EDMA_TPTC0 EVE3_EDMA_TPTC1 EVE4_EDMA_TPTC1
Description	TC Status Register		
Type	R		

31:13	12:11	10:9	8	7	6:4	3	2	1	0
RESERVED	DFSTRTPTR	RESERVED	ACTV	RESERVED	DSTACTV	RESERVED	WSACTV	SRCACTV	PROGBUSY

Bits	Field Name	Description	Type	Reset
31:13	RESERVED	Reserved	R Return 0's	0x0
12:11	DFSTRTPTR	Dst FIFO Start Pointer Represents the offset to the head entry of Dst Register FIFO, in units of *entries*. Legal values = 0x0 to 0x3	R	0x0
10:9	RESERVED	Reserved	R Return 0's	0x0
8	ACTV	Channel Active Channel Active is a logical-OR of each of the *BUSY/ACTV signals. The ACTV bit must remain high through the life of a TR. 0x0: Channel is idle 0x1: Channel is busy	R	0x1
7	RESERVED	Reserved	R Return 0's	0x0
6:4	DSTACTV	Destination Active State Specifies the number of TRs that are resident in the Dst Register FIFO at a given instant. Legal values are constrained by the DSTREGDEPTH parameter.	R	0x0
3	RESERVED	Reserved	R Return 0's	0x0
2	WSACTV	Write Status Active 0x0: Write status is not pending. Write status has been received for all previously issued write commands. 0x1: Write Status is pending. Write status has not been received for all previously issued write commands.	R	0x0

Bits	Field Name	Description	Type	Reset
1	SRCACTV	Source Active State 0x0: Source Active set is idle. Any TR written to Prog Set will immediately transition to Source Active set as long as the Dst FIFO Set is not full (DSTFULL == 1). 0x1: Source Active set is busy either performing read transfers or waiting to perform read transfers for current Transfer Request.	R	0x0
0	PROGBUSY	Program Register Set Busy 0x0: Program set idle and is available for programming. 0x1: Program set busy. User should poll for PROGBUSY equal to '0' prior to re-programming the Program Register set.	R	0x0

Table 16-356. Register Call Summary for Register EDMA_TPTCn_TCSTAT

Enhanced DMA

- Debug Features: [0] [1] [2] [3] [4] [5] [6] [7] [8] [9] [10] [11] [12] [13]
- EDMA Register Summary: [14] [15] [16] [17] [18] [19] [20] [21]

Table 16-357. EDMA_TPTCn_INTSTAT

Address Offset	0x0000 0104		
Physical Address	0x4340 0104 0x4350 0104 0x40D0 5104 0x40D0 6104 0x4150 5104 0x4150 6104 0x01D0 5104 0x01D0 6104 0x4208 6104 0x4218 6104 0x4208 7104 0x4218 7104 0x4228 6104 0x4238 6104 0x4228 7104 0x4238 7104	**Instance**	SYS_EDMA_TPTC0 SYS_EDMA_TPTC1 DSP1_EDMA_TPTC0 DSP1_EDMA_TPTC1 DSP2_EDMA_TPTC0 DSP2_EDMA_TPTC1 DSP_EDMA_TPTC0 DSP_EDMA_TPTC1 EVE1_EDMA_TPTC0 EVE2_EDMA_TPTC0 EVE1_EDMA_TPTC1 EVE2_EDMA_TPTC1 EVE3_EDMA_TPTC0 EVE4_EDMA_TPTC0 EVE3_EDMA_TPTC1 EVE4_EDMA_TPTC1
Description	Interrupt Status Register		
Type	R		

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

Bits	Field Name	Description	Type	Reset
31:2	RESERVED	Reserved	R Return 0's	0x0
1	TRDONE	TR Done Event Status 0x0: Condition not detected. 0x1: Set when TC has completed a Transfer Request. TRDONE should be set when the write status is returned for the final write of a TR. Cleared when write '1' to INTCLR.TRDONE register bit.	R	0x0

Bits	Field Name	Description	Type	Reset
0	PROGEMPTY	Program Set Empty Event Status 0x0: Condition not detected 0x1: Set when Program Register set transitions to empty state. Cleared when write '1' to EDMA_TPTCn_INTCLR[0] PROGEMPTY register bit.	R	0x0

Table 16-358. Register Call Summary for Register EDMA_TPTCn_INTSTAT

Enhanced DMA
- EDMA Register Summary: [0] [1] [2] [3] [4] [5] [6] [7]
- EDMA Register Description: [8]

Table 16-359. EDMA_TPTCn_INTEN

Address Offset	0x0000 0108		
Physical Address	0x4340 0108	**Instance**	SYS_EDMA_TPTC0
	0x4350 0108		SYS_EDMA_TPTC1
	0x40D0 5108		DSP1_EDMA_TPTC0
	0x40D0 6108		DSP1_EDMA_TPTC1
	0x4150 5108		DSP2_EDMA_TPTC0
	0x4150 6108		DSP2_EDMA_TPTC1
	0x01D0 5108		DSP_EDMA_TPTC0
	0x01D0 6108		DSP_EDMA_TPTC1
	0x4208 6108		EVE1_EDMA_TPTC0
	0x4218 6108		EVE2_EDMA_TPTC0
	0x4208 7108		EVE1_EDMA_TPTC1
	0x4218 7108		EVE2_EDMA_TPTC1
	0x4228 6108		EVE3_EDMA_TPTC0
	0x4238 6108		EVE4_EDMA_TPTC0
	0x4228 7108		EVE3_EDMA_TPTC1
	0x4238 7108		EVE4_EDMA_TPTC1
Description	Interrupt Enable Register		
Type	RW		

31:2	1	0
RESERVED	TRDONE	PROGEMPTY

Bits	Field Name	Description	Type	Reset
31:2	RESERVED	Reserved	R	0x0
1	TRDONE	TR Done Event Enable 0x0: TRDONE Event is disabled. 0x1: TRDONE Event is enabled, and contributes to interrupt generation	RW	0x0
0	PROGEMPTY	Program Set Empty Event Enable 0x0: PROGEMPTY Event is disabled. 0x1: PROGEMPTY Event is enabled, and contributes to interrupt generation	RW	0x0

Table 16-360. Register Call Summary for Register EDMA_TPTCn_INTEN

Enhanced DMA
- EDMA Register Summary: [0] [1] [2] [3] [4] [5] [6] [7]

Table 16-361. EDMA_TPTCn_INTCLR

Address Offset	0x0000 010C		
Physical Address	0x4340 010C 0x4350 010C 0x40D0 510C 0x40D0 610C 0x4150 510C 0x4150 610C 0x01D0 510C 0x01D0 610C 0x4208 610C 0x4218 610C 0x4208 710C 0x4218 710C 0x4228 610C 0x4238 610C 0x4228 710C 0x4238 710C	**Instance**	SYS_EDMA_TPTC0 SYS_EDMA_TPTC1 DSP1_EDMA_TPTC0 DSP1_EDMA_TPTC1 DSP2_EDMA_TPTC0 DSP2_EDMA_TPTC1 DSP_EDMA_TPTC0 DSP_EDMA_TPTC1 EVE1_EDMA_TPTC0 EVE2_EDMA_TPTC0 EVE1_EDMA_TPTC1 EVE2_EDMA_TPTC1 EVE3_EDMA_TPTC0 EVE4_EDMA_TPTC0 EVE3_EDMA_TPTC1 EVE4_EDMA_TPTC1
Description	Interrupt Clear Register		
Type	W		

31–2	1	0
RESERVED	TRDONE	PROGEMPTY

Bits	Field Name	Description	Type	Reset
31:2	RESERVED	Reserved	R	0x0
1	TRDONE	TR Done Event Clear Write 0x0: have no effect. Write 0x1: Clear	W	0x0
0	PROGEMPTY	Program Set Empty Event Clear Write 0x0: have no effect. Write 0x1: Clear	W	0x0

Table 16-362. Register Call Summary for Register EDMA_TPTCn_INTCLR

Enhanced DMA

- EDMA Register Summary: [0] [1] [2] [3] [4] [5] [6] [7]
- EDMA Register Description: [8]

Table 16-363. EDMA_TPTCn_INTCMD

Address Offset	0x0000 0110		
Physical Address	0x4340 0110 0x4350 0110 0x40D0 5110 0x40D0 6110 0x4150 5110 0x4150 6110 0x01D0 5110 0x01D0 6110 0x4208 6110 0x4218 6110 0x4208 7110 0x4218 7110 0x4228 6110 0x4238 6110 0x4228 7110 0x4238 7110	**Instance**	SYS_EDMA_TPTC0 SYS_EDMA_TPTC1 DSP1_EDMA_TPTC0 DSP1_EDMA_TPTC1 DSP2_EDMA_TPTC0 DSP2_EDMA_TPTC1 DSP_EDMA_TPTC0 DSP_EDMA_TPTC1 EVE1_EDMA_TPTC0 EVE2_EDMA_TPTC0 EVE1_EDMA_TPTC1 EVE2_EDMA_TPTC1 EVE3_EDMA_TPTC0 EVE4_EDMA_TPTC0 EVE3_EDMA_TPTC1 EVE4_EDMA_TPTC1
Description	Interrupt Command Register		
Type	W		

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

Bits	Field Name	Description	Type	Reset
31:2	RESERVED	Reserved	R	0x0
1	SET	Set TPTC interrupt Write 0x0: have no affect. Write 0x1: SET causes TPTC interrupt to be pulsed unconditionally	W	0x0
0	EVAL	Evaluate state of TPTC interrupt Write 0x0: have no affect. 0x1: causes TPTC interrupt to be pulsed if any of the EDMA_TPTCn_INTSTAT bits are set to '1'.	W	0x0

Table 16-364. Register Call Summary for Register EDMA_TPTCn_INTCMD

Enhanced DMA

- EDMA Register Summary: [0] [1] [2] [3] [4] [5] [6] [7]

Table 16-365. EDMA_TPTCn_ERRSTAT

Address Offset	0x0000 0120		
Physical Address	0x4340 0120 0x4350 0120 0x40D0 5120 0x40D0 6120 0x4150 5120 0x4150 6120 0x01D0 5120 0x01D0 6120 0x4208 6120 0x4218 6120 0x4208 7120 0x4218 7120 0x4228 6120 0x4238 6120 0x4228 7120 0x4238 7120	**Instance**	SYS_EDMA_TPTC0 SYS_EDMA_TPTC1 DSP1_EDMA_TPTC0 DSP1_EDMA_TPTC1 DSP2_EDMA_TPTC0 DSP2_EDMA_TPTC1 DSP_EDMA_TPTC0 DSP_EDMA_TPTC1 EVE1_EDMA_TPTC0 EVE2_EDMA_TPTC0 EVE1_EDMA_TPTC1 EVE2_EDMA_TPTC1 EVE3_EDMA_TPTC0 EVE4_EDMA_TPTC0 EVE3_EDMA_TPTC1 EVE4_EDMA_TPTC1

Table 16-365. EDMA_TPTCn_ERRSTAT (continued)

Description	Error Status Register
Type	R

31–4	3	2	1	0
RESERVED	MMRAERR	TRERR	RESERVED	BUSERR

Bits	Field Name	Description	Type	Reset
31:4	RESERVED	Reserved	R Return 0's	0x0
3	MMRAERR	MR Address Error 0x0: Condition not detected 0x1: User attempted to read or write to invalid address configuration mMemory map. (Is only be set for non-emulation accesses). No additional error information is recorded.	R	0x0
2	TRERR	TR Error: TR detected that violates FIFO Mode transfer (SAM or DAM is '1') alignment rules or has ACNT or BCNT == 0. No additional error information is recorded.	R	0x0
1	RESERVED	Reserved	R Return 0's	0x0
0	BUSERR	Bus Error Event 0x0: Condition not detected. 0x1: TC has detected an error code on the write response bus or read response bus. Error information is stored in Error Details Register (EDMA_TPTCn_ERRDET).	R	0x0

Table 16-366. Register Call Summary for Register EDMA_TPTCn_ERRSTAT

Enhanced DMA

- EDMA Register Summary: [0] [1] [2] [3] [4] [5] [6] [7]
- EDMA Register Description: [8] [9] [10] [11] [12] [13] [14] [15] [16] [17]

Table 16-367. EDMA_TPTCn_ERREN

Address Offset	0x0000 0124		
Physical Address	0x4340 0124	**Instance**	SYS_EDMA_TPTC0
	0x4350 0124		SYS_EDMA_TPTC1
	0x40D0 5124		DSP1_EDMA_TPTC0
	0x40D0 6124		DSP1_EDMA_TPTC1
	0x4150 5124		DSP2_EDMA_TPTC0
	0x4150 6124		DSP2_EDMA_TPTC1
	0x01D0 5124		DSP_EDMA_TPTC0
	0x01D0 6124		DSP_EDMA_TPTC1
	0x4208 6124		EVE1_EDMA_TPTC0
	0x4218 6124		EVE2_EDMA_TPTC0
	0x4208 7124		EVE1_EDMA_TPTC1
	0x4218 7124		EVE2_EDMA_TPTC1
	0x4228 6124		EVE3_EDMA_TPTC0
	0x4238 6124		EVE4_EDMA_TPTC0
	0x4228 7124		EVE3_EDMA_TPTC1
	0x4238 7124		EVE4_EDMA_TPTC1
Description	Error Enable Register		

Table 16-367. EDMA_TPTCn_ERREN (continued)

Type	RW

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

Bits	Field Name	Description	Type	Reset
31:4	RESERVED	Reserved	R Return 0's	0x0
3	MMRAERR	Interrupt enable for EDMA_TPTCn_ERRSTAT[3] MMRAERR 0x0: BUSERR is disabled 0x1: MMRAERR is enabled, and contributes to the TPTC error interrupt generation.	RW	0x0
2	TRERR	Interrupt enable for EDMA_TPTCn_ERRSTAT[2] TRERR 0x0: BUSERR is disabled. 0x1: TRERR is enabled, and contributes to the TPTC error interrupt generation.	RW	0x0
1	RESERVED	Reserved	R Return 0's	0x0
0	BUSERR	Interrupt enable for EDMA_TPTCn_ERRSTAT[0] BUSERR 0x0: BUSERR is disabled. 0x1: BUSERR is enabled, and contributes to the TPTC error interrupt generation.	RW	0x0

Table 16-368. Register Call Summary for Register EDMA_TPTCn_ERREN

Enhanced DMA

- EDMA Register Summary: [0] [1] [2] [3] [4] [5] [6] [7]

Table 16-369. EDMA_TPTCn_ERRCLR

Address Offset	0x0000 0128		
Physical Address	0x4340 0128 0x4350 0128 0x40D0 5128 0x40D0 6128 0x4150 5128 0x4150 6128 0x01D0 5128 0x01D0 6128 0x4208 6128 0x4218 6128 0x4208 7128 0x4218 7128 0x4228 6128 0x4238 6128 0x4228 7128 0x4238 7128	**Instance**	SYS_EDMA_TPTC0 SYS_EDMA_TPTC1 DSP1_EDMA_TPTC0 DSP1_EDMA_TPTC1 DSP2_EDMA_TPTC0 DSP2_EDMA_TPTC1 DSP_EDMA_TPTC0 DSP_EDMA_TPTC1 EVE1_EDMA_TPTC0 EVE2_EDMA_TPTC0 EVE1_EDMA_TPTC1 EVE2_EDMA_TPTC1 EVE3_EDMA_TPTC0 EVE4_EDMA_TPTC0 EVE3_EDMA_TPTC1 EVE4_EDMA_TPTC1
Description	Error Clear Register		
Type	W		

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

Bits	Field Name	Description	Type	Reset
31:4	RESERVED	Reserved	R Return 0's	0x0
3	MMRAERR	Interrupt clear for EDMA_TPTCn_ERRSTAT[3] MMRAERR Write 0x0: have no effect Write 0x1: to clear EDMA_TPTCn_ERRSTAT[3] MMRAERR bit. Write of '1' to EDMA_TPTCn_ERRCLR[3] MMRAERR does not clear the ERRDET register.	W	0x0
2	TRERR	Interrupt clear for EDMA_TPTCn_ERRSTAT[2] TRERR Write 0x0: have no effect Write 0x1: to clear EDMA_TPTCn_ERRSTAT[2] TRERR bit. Write of '1' to EDMA_TPTCn_ERRCLR[2] TRERR does not clear the ERRDET register.	W	0x0
1	RESERVED	Reserved	R Return 0's	0x0
0	BUSERR	Interrupt clear for EDMA_TPTCn_ERRSTAT[0] BUSERR Write 0x0: have no effect Write 0x1: to clear EDMA_TPTCn_ERRSTAT[0] BUSERR bit Write of '1' to EDMA_TPTCn_ERRCLR[0] BUSERR clears the ERRDET register.	W	0x0

Table 16-370. Register Call Summary for Register EDMA_TPTCn_ERRCLR

Enhanced DMA
- EDMA Register Summary: [0] [1] [2] [3] [4] [5] [6] [7]
- EDMA Register Description: [8] [9] [10]

Table 16-371. EDMA_TPTCn_ERRDET

Address Offset	0x0000 012C		
Physical Address	0x4340 012C 0x4350 012C 0x40D0 512C 0x40D0 612C 0x4150 512C 0x4150 612C 0x01D0 512C 0x01D0 612C 0x4208 612C 0x4218 612C 0x4208 712C 0x4218 712C 0x4228 612C 0x4238 612C 0x4228 712C 0x4238 712C	**Instance**	SYS_EDMA_TPTC0 SYS_EDMA_TPTC1 DSP1_EDMA_TPTC0 DSP1_EDMA_TPTC1 DSP2_EDMA_TPTC0 DSP2_EDMA_TPTC1 DSP_EDMA_TPTC0 DSP_EDMA_TPTC1 EVE1_EDMA_TPTC0 EVE2_EDMA_TPTC0 EVE1_EDMA_TPTC1 EVE2_EDMA_TPTC1 EVE3_EDMA_TPTC0 EVE4_EDMA_TPTC0 EVE3_EDMA_TPTC1 EVE4_EDMA_TPTC1
Description	Error Details Register		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED														TCCHEN	TCINTEN	RESERVED		TCC						RESERVED				STAT			

Bits	Field Name	Description	Type	Reset
31:18	RESERVED	Reserved	R Return 0's	0x0
17	TCCHEN	Contains the EDMA_TPCC_OPT_n[17] TCCHEN value programmed by the user for the Read or Write transaction that resulted in an error.	R	0x0
16	TCINTEN	Contains the EDMA_TPCC_OPT_n[16] TCINTEN value programmed by the user for the Read or Write transaction that resulted in an error.	R	0x0
15:14	RESERVED	Reserved	R Return 0's	0x0
13:8	TCC	Transfer Complete Code: Contains the EDMA_TPCC_OPT_n[13:8] TCC value programmed by the user for the Read or Write transaction that resulted in an error.	R	0x0
7:4	RESERVED	Reserved	R Return 0's	0x0
3:0	STAT	Transaction Status: Stores the non-zero status/error code that was detected on the read status or write status bus. MS-bit effectively serves as the read vs. write error code. If read status and write status are returned on the same cycle, then the TC chooses non-zero version. If both are non-zero then write status is treated as higher priority. Encoding of errors matches the CBA spec and is summarized here: 0xF =	R	0x0

Table 16-372. Register Call Summary for Register EDMA_TPTCn_ERRDET

Enhanced DMA

- EDMA Register Summary: [0] [1] [2] [3] [4] [5] [6] [7]
- EDMA Register Description: [8]

Table 16-373. EDMA_TPTCn_ERRCMD

Address Offset	0x0000 0130		
Physical Address	0x4340 0130 0x4350 0130 0x40D0 5130 0x40D0 6130 0x4150 5130 0x4150 6130 0x01D0 5130 0x01D0 6130 0x4208 6130 0x4218 6130 0x4208 7130 0x4218 7130 0x4228 6130 0x4238 6130 0x4228 7130 0x4238 7130	**Instance**	SYS_EDMA_TPTC0 SYS_EDMA_TPTC1 DSP1_EDMA_TPTC0 DSP1_EDMA_TPTC1 DSP2_EDMA_TPTC0 DSP2_EDMA_TPTC1 DSP_EDMA_TPTC0 DSP_EDMA_TPTC1 EVE1_EDMA_TPTC0 EVE2_EDMA_TPTC0 EVE1_EDMA_TPTC1 EVE2_EDMA_TPTC1 EVE3_EDMA_TPTC0 EVE4_EDMA_TPTC0 EVE3_EDMA_TPTC1 EVE4_EDMA_TPTC1
Description	Error Command Register		
Type	W		

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

Bits	Field Name	Description	Type	Reset
31:2	RESERVED	Reserved	R	0x0
1	SET	Set TPTC error interrupt Write 0x0: have no affect Write 0x1: to SET causes TPTC error interrupt to be pulsed unconditionally.	W	0x0
0	EVAL	Evaluate state of TPTC error interrupt Write of '1' Write 0x0: have no affect Write 0x1: to EVAL causes TPTC error interrupt to be pulsed if any of the EDMA_TPTCn_ERRSTAT bits are set to '1'.	W	0x0

Table 16-374. Register Call Summary for Register EDMA_TPTCn_ERRCMD

Enhanced DMA
- EDMA Register Summary: [0] [1] [2] [3] [4] [5] [6] [7]

Table 16-375. EDMA_TPTCn_RDRATE

Address Offset	0x0000 0140		
Physical Address	0x4340 0140 0x4350 0140 0x40D0 5140 0x40D0 6140 0x4150 5140 0x4150 6140 0x01D0 5140 0x01D0 6140 0x4208 6140 0x4218 6140 0x4208 7140 0x4218 7140 0x4228 6140 0x4238 6140 0x4228 7140 0x4238 7140	**Instance**	SYS_EDMA_TPTC0 SYS_EDMA_TPTC1 DSP1_EDMA_TPTC0 DSP1_EDMA_TPTC1 DSP2_EDMA_TPTC0 DSP2_EDMA_TPTC1 DSP_EDMA_TPTC0 DSP_EDMA_TPTC1 EVE1_EDMA_TPTC0 EVE2_EDMA_TPTC0 EVE1_EDMA_TPTC1 EVE2_EDMA_TPTC1 EVE3_EDMA_TPTC0 EVE4_EDMA_TPTC0 EVE3_EDMA_TPTC1 EVE4_EDMA_TPTC1
Description	Read Rate Register		
Type	RW		

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

Bits	Field Name	Description	Type	Reset
31:3	RESERVED	Reserved	R	0x0
2:0	RDRATE	Read Rate Control: Controls the number of cycles between read commands. This is a global setting that applies to all TRs for this TC.	RW	0x0

Table 16-376. Register Call Summary for Register EDMA_TPTCn_RDRATE

Enhanced DMA
- Architecture Details: [0] [1] [2]
- EDMA Register Summary: [3] [4] [5] [6] [7] [8] [9] [10]

Table 16-377. EDMA_TPTCn_POPT

Address Offset	0x0000 0200		
Physical Address	0x4340 0200 0x4350 0200 0x40D0 5200 0x40D0 6200 0x4150 5200 0x4150 6200 0x01D0 5200 0x01D0 6200 0x4208 6200 0x4218 6200 0x4208 7200 0x4218 7200 0x4228 6200 0x4238 6200 0x4228 7200 0x4238 7200	**Instance**	SYS_EDMA_TPTC0 SYS_EDMA_TPTC1 DSP1_EDMA_TPTC0 DSP1_EDMA_TPTC1 DSP2_EDMA_TPTC0 DSP2_EDMA_TPTC1 DSP_EDMA_TPTC0 DSP_EDMA_TPTC1 EVE1_EDMA_TPTC0 EVE2_EDMA_TPTC0 EVE1_EDMA_TPTC1 EVE2_EDMA_TPTC1 EVE3_EDMA_TPTC0 EVE4_EDMA_TPTC0 EVE3_EDMA_TPTC1 EVE4_EDMA_TPTC1
Description	Program Set Options		
Type	RW		

31 - 23	22	21	20	19 - 18	17 - 12	11	10 - 8	7	6 - 4	3 - 2	1	0
RESERVED	TCCHEN	RESERVED	TCINTEN	RESERVED	TCC	RESERVED	FWID	RESERVED	PRI	RESERVED	DAM	SAM

Bits	Field Name	Description	Type	Reset
31:23	RESERVED	Reserved	R	0x0
22	TCCHEN	Transfer complete chaining enable 0x0: Transfer complete chaining is disabled. 0x1: Transfer complete chaining is enabled.	RW	0x0
21	RESERVED	Reserved	R	0x0
20	TCINTEN	Transfer complete interrupt enable 0x0: Transfer complete interrupt is disabled. 0x1: Transfer complete interrupt is enabled.	RW	0x0
19:18	RESERVED	Reserved	R	0x0
17:12	TCC	Transfer Complete Code: The 6-bit code is used to set the relevant bit in CER or EDMA_TPCC_IPR of the TPCC module.	RW	0x0
11	RESERVED	Reserved	R	0x0
10:8	FWID	FIFO width control: Applies if either SAM or DAM is set to FIFO mode.	RW	0x0
7	RESERVED	Reserved	R	0x0
6:4	PRI	Transfer Priority: 0x0: Priority 0 - Highest priority 0x1: Priority 1 ... 0x7: Priority 7 - Lowest priority	RW	0x0
3:2	RESERVED	Reserved	R	0x0
1	DAM	Destination Address Mode within an array 0x0: INCR, Destination addressing within an array increments. 0x1: FIFO, Destination addressing within an array wraps around upon reaching FIFO width.	RW	0x0

Bits	Field Name	Description	Type	Reset
0	SAM	Source Address Mode within an array 0x0: INCR, Source addressing within an array increments. 0x1: FIFO, Source addressing within an array wraps around upon reaching FIFO width.	RW	0x0

Table 16-378. Register Call Summary for Register EDMA_TPTCn_POPT

Enhanced DMA

- EDMA Register Summary: [0] [1] [2] [3] [4] [5] [6] [7]

Table 16-379. EDMA_TPTCn_PSRC

Address Offset	0x0000 0204		
Physical Address	0x4340 0204 0x4350 0204 0x40D0 5204 0x40D0 6204 0x4150 5204 0x4150 6204 0x01D0 5204 0x01D0 6204 0x4208 6204 0x4218 6204 0x4208 7204 0x4218 7204 0x4228 6204 0x4238 6204 0x4228 7204 0x4238 7204	**Instance**	SYS_EDMA_TPTC0 SYS_EDMA_TPTC1 DSP1_EDMA_TPTC0 DSP1_EDMA_TPTC1 DSP2_EDMA_TPTC0 DSP2_EDMA_TPTC1 DSP_EDMA_TPTC0 DSP_EDMA_TPTC1 EVE1_EDMA_TPTC0 EVE2_EDMA_TPTC0 EVE1_EDMA_TPTC1 EVE2_EDMA_TPTC1 EVE3_EDMA_TPTC0 EVE4_EDMA_TPTC0 EVE3_EDMA_TPTC1 EVE4_EDMA_TPTC1
Description	Program Set Source Address		
Type	RW		

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

Bits	Field Name	Description	Type	Reset
31:0	SADDR	Source address for Program Register Set	RW	0x0

Table 16-380. Register Call Summary for Register EDMA_TPTCn_PSRC

Enhanced DMA

- Architecture Details: [0] [1]
- EDMA Register Summary: [2] [3] [4] [5] [6] [7] [8] [9]
- EDMA Register Description: [10]

Table 16-381. EDMA_TPTCn_PCNT

Address Offset	0x0000 0208		
Physical Address	0x4340 0208 0x4350 0208 0x40D0 5208 0x40D0 6208 0x4150 5208 0x4150 6208 0x01D0 5208 0x01D0 6208 0x4208 6208 0x4218 6208 0x4208 7208 0x4218 7208 0x4228 6208 0x4238 6208 0x4228 7208 0x4238 7208	**Instance**	SYS_EDMA_TPTC0 SYS_EDMA_TPTC1 DSP1_EDMA_TPTC0 DSP1_EDMA_TPTC1 DSP2_EDMA_TPTC0 DSP2_EDMA_TPTC1 DSP_EDMA_TPTC0 DSP_EDMA_TPTC1 EVE1_EDMA_TPTC0 EVE2_EDMA_TPTC0 EVE1_EDMA_TPTC1 EVE2_EDMA_TPTC1 EVE3_EDMA_TPTC0 EVE4_EDMA_TPTC0 EVE3_EDMA_TPTC1 EVE4_EDMA_TPTC1
Description	Program Set Count		
Type	RW		

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

Bits	Field Name	Description	Type	Reset
31:16	BCNT	B-Dimension count. Number of arrays to be transferred, where each array is ACNT in length.	RW	0x0
15:0	ACNT	A-Dimension count. Number of bytes to be transferred in first dimension.	RW	0x0

Table 16-382. Register Call Summary for Register EDMA_TPTCn_PCNT

Enhanced DMA

- Architecture Details: [0] [1] [2] [3] [4] [5]
- EDMA Register Summary: [6] [7] [8] [9] [10] [11] [12] [13]
- EDMA Register Description: [14] [15] [16] [17] [18] [19]

Table 16-383. EDMA_TPTCn_PDST

Address Offset	0x0000 020C		
Physical Address	0x4340 020C 0x4350 020C 0x40D0 520C 0x40D0 620C 0x4150 520C 0x4150 620C 0x01D0 520C 0x01D0 620C 0x4208 620C 0x4218 620C 0x4208 720C 0x4218 720C 0x4228 620C 0x4238 620C 0x4228 720C 0x4238 720C	**Instance**	SYS_EDMA_TPTC0 SYS_EDMA_TPTC1 DSP1_EDMA_TPTC0 DSP1_EDMA_TPTC1 DSP2_EDMA_TPTC0 DSP2_EDMA_TPTC1 DSP_EDMA_TPTC0 DSP_EDMA_TPTC1 EVE1_EDMA_TPTC0 EVE2_EDMA_TPTC0 EVE1_EDMA_TPTC1 EVE2_EDMA_TPTC1 EVE3_EDMA_TPTC0 EVE4_EDMA_TPTC0 EVE3_EDMA_TPTC1 EVE4_EDMA_TPTC1
Description	Program Set Destination Address		
Type	RW		

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

Bits	Field Name	Description	Type	Reset
31:0	DADDR	Destination address for Program Register Set	RW	0x0

Table 16-384. Register Call Summary for Register EDMA_TPTCn_PDST

Enhanced DMA
- EDMA Register Summary: [0] [1] [2] [3] [4] [5] [6] [7]
- EDMA Register Description: [8]

Table 16-385. EDMA_TPTCn_PBIDX

Address Offset	0x0000 0210		
Physical Address	0x4340 0210	**Instance**	SYS_EDMA_TPTC0
	0x4350 0210		SYS_EDMA_TPTC1
	0x40D0 5210		DSP1_EDMA_TPTC0
	0x40D0 6210		DSP1_EDMA_TPTC1
	0x4150 5210		DSP2_EDMA_TPTC0
	0x4150 6210		DSP2_EDMA_TPTC1
	0x01D0 5210		DSP_EDMA_TPTC0
	0x01D0 6210		DSP_EDMA_TPTC1
	0x4208 6210		EVE1_EDMA_TPTC0
	0x4218 6210		EVE2_EDMA_TPTC0
	0x4208 7210		EVE1_EDMA_TPTC1
	0x4218 7210		EVE2_EDMA_TPTC1
	0x4228 6210		EVE3_EDMA_TPTC0
	0x4238 6210		EVE4_EDMA_TPTC0
	0x4228 7210		EVE3_EDMA_TPTC1
	0x4238 7210		EVE4_EDMA_TPTC1
Description	Program Set B-Dim Idx		
Type	RW		

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

Bits	Field Name	Description	Type	Reset
31:16	DBIDX	Dest B-Idx for Program Register Set: B-Idx offset between Destination arrays: Represents the offset in bytes between the starting address of each destination array (recall that there are BCNT arrays of ACNT elements). DBIDX is always used, regardless of whether DAM is Increment or FIFO mode.	RW	0x0
15:0	SBIDX	Source B-Idx for Program Register Set: B-Idx offset between Source arrays: Represents the offset in bytes between the starting address of each source array (recall that there are BCNT arrays of ACNT elements). SBIDX is always used, regardless of whether SAM is Increment or FIFO mode.	RW	0x0

Table 16-386. Register Call Summary for Register EDMA_TPTCn_PBIDX

Enhanced DMA
- Architecture Details: [0] [1]
- EDMA Register Summary: [2] [3] [4] [5] [6] [7] [8] [9]
- EDMA Register Description: [10] [11] [12] [13]

Table 16-387. EDMA_TPTCn_PMPPRXY

Address Offset	0x0000 0214		
Physical Address	0x4340 0214 0x4350 0214 0x40D0 5214 0x40D0 6214 0x4150 5214 0x4150 6214 0x01D0 5214 0x01D0 6214 0x4208 6214 0x4218 6214 0x4208 7214 0x4218 7214 0x4228 6214 0x4238 6214 0x4228 7214 0x4238 7214	**Instance**	SYS_EDMA_TPTC0 SYS_EDMA_TPTC1 DSP1_EDMA_TPTC0 DSP1_EDMA_TPTC1 DSP2_EDMA_TPTC0 DSP2_EDMA_TPTC1 DSP_EDMA_TPTC0 DSP_EDMA_TPTC1 EVE1_EDMA_TPTC0 EVE2_EDMA_TPTC0 EVE1_EDMA_TPTC1 EVE2_EDMA_TPTC1 EVE3_EDMA_TPTC0 EVE4_EDMA_TPTC0 EVE3_EDMA_TPTC1 EVE4_EDMA_TPTC1
Description	Program Set Memory Protect Proxy		
Type	R		

31:9	8	7:4	3:0
RESERVED	PRIV	RESERVED	PRIVID

Bits	Field Name	Description	Type	Reset
31:9	RESERVED	Reserved	R Return 0's	0x0
8	PRIV	Privilege Level 0x0: User level privilege 0x1: Supervisor level privilege EDMA_TPTCn_PMPPRXY.PRIV is always updated with the value from the configuration bus privilege field on any/every write to Program Set BIDX Register (trigger register). The PRIV value for the SA Set and DF Set are copied from the value in the Program set along with the remainder of the parameter values. The privilege ID is issued on the VBusM read and write command bus such that the target endpoints can perform mMemory protection checks based on the PRIV of the external host that sets up the DMA transaction.	R	0x0
7:4	RESERVED	Reserved	R Return 0's	0x0
3:0	PRIVID	Privilege ID: EDMA_TPTCn_PMPPRXY.PRIVID is always updated with the value from configuration bus privilege ID field on any/every write to Program Set BIDX Register (trigger register). The PRIVID value for the SA Set and DF Set are copied from the value in the Program set along with the remainder of the parameter values. The privilege ID is issued on the VBusM read and write command bus such that the target endpoints can perform mMemory protection checks based on the privid of the external host that sets up the DMA transaction.	R	0x0

Table 16-388. Register Call Summary for Register EDMA_TPTCn_PMPPRXY

Enhanced DMA
- EDMA Register Summary: [0] [1] [2] [3] [4] [5] [6] [7]
- EDMA Register Description: [8] [9]

Table 16-389. EDMA_TPTCn_SAOPT

Address Offset	0x0000 0240		
Physical Address	0x4340 0240 0x4350 0240 0x40D0 5240 0x40D0 6240 0x4150 5240 0x4150 6240 0x01D0 5240 0x01D0 6240 0x4208 6240 0x4218 6240 0x4208 7240 0x4218 7240 0x4228 6240 0x4238 6240 0x4228 7240 0x4238 7240	**Instance**	SYS_EDMA_TPTC0 SYS_EDMA_TPTC1 DSP1_EDMA_TPTC0 DSP1_EDMA_TPTC1 DSP2_EDMA_TPTC0 DSP2_EDMA_TPTC1 DSP_EDMA_TPTC0 DSP_EDMA_TPTC1 EVE1_EDMA_TPTC0 EVE2_EDMA_TPTC0 EVE1_EDMA_TPTC1 EVE2_EDMA_TPTC1 EVE3_EDMA_TPTC0 EVE4_EDMA_TPTC0 EVE3_EDMA_TPTC1 EVE4_EDMA_TPTC1
Description	Source Actve Set Options		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED									TCCHEN	RESERVED	TCINTEN	RESERVED		TCC						RESERVED	FWID			RESERVED	PRI			RESERVED		DAM	SAM

Bits	Field Name	Description	Type	Reset
31:23	RESERVED	Reserved	R Return 0's	0x0
22	TCCHEN	Transfer complete chaining enable 0x0: Transfer complete chaining is disabled. 0x1: Transfer complete chaining is enabled.	R	0x0
21	RESERVED	Reserved	R Return 0's	0x0
20	TCINTEN	Transfer complete interrupt enable 0x0: Transfer complete interrupt is disabled. 0x1: Transfer complete interrupt is enabled.	R	0x0
19:18	RESERVED	Reserved	R Return 0's	0x0
17:12	TCC	Transfer Complete Code The 6-bit code is used to set the relevant bit in EDMA_TPCC_CER or EDMA_TPCC_IPR of the TPCC module.	R	0x0
11	RESERVED	Reserved	R Return 0's	0x0
10:8	FWID	FIFO width control Applies if either SAM or DAM is set to FIFO mode.	R	0x0

Bits	Field Name	Description	Type	Reset
7	RESERVED	Reserved	R Return 0's	0x0
6:4	PRI	Transfer Priority 0x0: Priority 0 - Highest priority 0x1: Priority 1 ... 0x7: Priority 7 - Lowest priority	R	0x0
3:2	RESERVED	Reserved	R Return 0's	0x0
1	DAM	Destination Address Mode within an array 0x0: INCR, Destination addressing within an array increments. 0x1: FIFO, Destination addressing within an array wraps around upon reaching FIFO width.	R	0x0
0	SAM	Source Address Mode within an array 0x0: INCR, Source addressing within an array increments. 0X1: FIFO, Source addressing within an array wraps around upon reaching FIFO width.	R	0x0

Table 16-390. Register Call Summary for Register EDMA_TPTCn_SAOPT

Enhanced DMA

- EDMA Register Summary: [0] [1] [2] [3] [4] [5] [6] [7]

Table 16-391. EDMA_TPTCn_SASRC

Address Offset	0x0000 0244		
Physical Address	0x4340 0244 0x4350 0244 0x40D0 5244 0x40D0 6244 0x4150 5244 0x4150 6244 0x01D0 5244 0x01D0 6244 0x4208 6244 0x4218 6244 0x4208 7244 0x4218 7244 0x4228 6244 0x4238 6244 0x4228 7244 0x4238 7244	**Instance**	SYS_EDMA_TPTC0 SYS_EDMA_TPTC1 DSP1_EDMA_TPTC0 DSP1_EDMA_TPTC1 DSP2_EDMA_TPTC0 DSP2_EDMA_TPTC1 DSP_EDMA_TPTC0 DSP_EDMA_TPTC1 EVE1_EDMA_TPTC0 EVE2_EDMA_TPTC0 EVE1_EDMA_TPTC1 EVE2_EDMA_TPTC1 EVE3_EDMA_TPTC0 EVE4_EDMA_TPTC0 EVE3_EDMA_TPTC1 EVE4_EDMA_TPTC1
Description	Source Active Set Source Address		
Type	R		

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

Bits	Field Name	Description	Type	Reset
31:0	SADDR	Source address for Source Active Register Set: Initial value is copied from EDMA_TPTCn_PSRC.SADDR. TC updates value according to source addressing mode (EDMA_TPCC_OPT_n.SAM) and/or source index value (BIDX.SBIDX) after each read command is issued. When a TR is complete, the final value should be the address of the last read command issued.	R	0x0

Table 16-392. Register Call Summary for Register EDMA_TPTCn_SASRC

Enhanced DMA

- EDMA Register Summary: [0] [1] [2] [3] [4] [5] [6] [7]

Table 16-393. EDMA_TPTCn_SACNT

Address Offset	0x0000 0248		
Physical Address	0x4340 0248 0x4350 0248 0x40D0 5248 0x40D0 6248 0x4150 5248 0x4150 6248 0x01D0 5248 0x01D0 6248 0x4208 6248 0x4218 6248 0x4208 7248 0x4218 7248 0x4228 6248 0x4238 6248 0x4228 7248 0x4238 7248	**Instance**	SYS_EDMA_TPTC0 SYS_EDMA_TPTC1 DSP1_EDMA_TPTC0 DSP1_EDMA_TPTC1 DSP2_EDMA_TPTC0 DSP2_EDMA_TPTC1 DSP_EDMA_TPTC0 DSP_EDMA_TPTC1 EVE1_EDMA_TPTC0 EVE2_EDMA_TPTC0 EVE1_EDMA_TPTC1 EVE2_EDMA_TPTC1 EVE3_EDMA_TPTC0 EVE4_EDMA_TPTC0 EVE3_EDMA_TPTC1 EVE4_EDMA_TPTC1
Description	Source Active Set Count		
Type	R		

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

Bits	Field Name	Description	Type	Reset
31:16	BCNT	B-Dimension count: Number of arrays to be transferred, where each array is ACNT in length. Count Remaining for Source Active Register Set. Represents the amount of data remaining to be read. Initial value is copied from EDMA_TPTCn_PCNT. TC decrements ACNT and BCNT as necessary after each read command is issued. Final value should be 0 when TR is complete.	R	0x0
15:0	ACNT	A-Dimension count: Number of bytes to be transferred in first dimension. Count Remaining for Source Active Register Set. Represents the amount of data remaining to be read. Initial value is copied from EDMA_TPTCn_PCNT. TC decrements ACNT and BCNT as necessary after each read command is issued. Final value should be 0 when TR is complete.	R	0x0

Table 16-394. Register Call Summary for Register EDMA_TPTCn_SACNT

Enhanced DMA

- EDMA Register Summary: [0] [1] [2] [3] [4] [5] [6] [7]

Table 16-395. EDMA_TPTCn_SADST

Address Offset	0x0000 024C		
Physical Address	0x4340 024C 0x4350 024C 0x40D0 524C 0x40D0 624C 0x4150 524C 0x4150 624C 0x01D0 524C 0x01D0 624C	**Instance**	SYS_EDMA_TPTC0 SYS_EDMA_TPTC1 DSP1_EDMA_TPTC0 DSP1_EDMA_TPTC1 DSP2_EDMA_TPTC0 DSP2_EDMA_TPTC1 DSP_EDMA_TPTC0 DSP_EDMA_TPTC1
Description	Source Active Destination Address Register Reserved, return 0x0 w/o AERROR		
Type	R		

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

Bits	Field Name	Description	Type	Reset
31:0	DADDR	Destination address is not applicable for Source Active Register Set. Reads return 0x0	R	0x0

Table 16-396. Register Call Summary for Register EDMA_TPTCn_SADST

Enhanced DMA

- Architecture Details: [0] [1]
- EDMA Register Summary: [2] [3] [4] [5] [6] [7] [8] [9]

Table 16-397. EDMA_TPTCn_SABIDX

Address Offset	0x0000 0250		
Physical Address	0x4340 0250 0x4350 0250 0x40D0 5250 0x40D0 6250 0x4150 5250 0x4150 6250 0x01D0 5250 0x01D0 6250 0x4208 6250 0x4218 6250 0x4208 7250 0x4218 7250 0x4228 6250 0x4238 6250 0x4228 7250 0x4238 7250	**Instance**	SYS_EDMA_TPTC0 SYS_EDMA_TPTC1 DSP1_EDMA_TPTC0 DSP1_EDMA_TPTC1 DSP2_EDMA_TPTC0 DSP2_EDMA_TPTC1 DSP_EDMA_TPTC0 DSP_EDMA_TPTC1 EVE1_EDMA_TPTC0 EVE2_EDMA_TPTC0 EVE1_EDMA_TPTC1 EVE2_EDMA_TPTC1 EVE3_EDMA_TPTC0 EVE4_EDMA_TPTC0 EVE3_EDMA_TPTC1 EVE4_EDMA_TPTC1
Description	Source Active Set B-Dim Idx		
Type	R		

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

Bits	Field Name	Description	Type	Reset
31:16	DBIDX	Destination B-Idx for Source Active Register Set. Value copied from EDMA_TPTCn_PBIDX: B-Idx offset between Destination arrays: Represents the offset in bytes between the starting address of each destination array (recall that there are BCNT arrays of ACNT elements). DBIDX is always used, regardless of whether DAM is Increment or FIFO mode.	R	0x0
15:0	SBIDX	Source B-Idx for Source Active Register Set. Value copied from EDMA_TPTCn_PBIDX: B-Idx offset between Source arrays: Represents the offset in bytes between the starting address of each source array (recall that there are BCNT arrays of ACNT elements). SBIDX is always used, regardless of whether SAM is Increment or FIFO mode.	R	0x0

Table 16-398. Register Call Summary for Register EDMA_TPTCn_SABIDX

Enhanced DMA

- EDMA Register Summary: [0] [1] [2] [3] [4] [5] [6] [7]

Table 16-399. EDMA_TPTCn_SAMPPRXY

Address Offset	0x0000 0254		
Physical Address	0x4340 0254	**Instance**	SYS_EDMA_TPTC0
	0x4350 0254		SYS_EDMA_TPTC1
	0x40D0 5254		DSP1_EDMA_TPTC0
	0x40D0 6254		DSP1_EDMA_TPTC1
	0x4150 5254		DSP2_EDMA_TPTC0
	0x4150 6254		DSP2_EDMA_TPTC1
	0x01D0 5254		DSP_EDMA_TPTC0
	0x01D0 6254		DSP_EDMA_TPTC1
	0x4208 6254		EVE1_EDMA_TPTC0
	0x4218 6254		EVE2_EDMA_TPTC0
	0x4208 7254		EVE1_EDMA_TPTC1
	0x4218 7254		EVE2_EDMA_TPTC1
	0x4228 6254		EVE3_EDMA_TPTC0
	0x4238 6254		EVE4_EDMA_TPTC0
	0x4228 7254		EVE3_EDMA_TPTC1
	0x4238 7254		EVE4_EDMA_TPTC1
Description	Source Active Set Mem Protect Proxy		
Type	R		

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

Bits	Field Name	Description	Type	Reset
31:9	RESERVED	Reserved	R Return 0's	0x0
8	PRIV	Privilege Level 0x0: User level privilege 0x1: Supervisor level privilege SAMPPRXY.PRIV is always updated with the value from the configuration bus privilege field on any/every write to Program Set BIDX Register (trigger register). The PRIV value for the SA Set and DF Set are copied from the value in the Program set along with the remainder of the parameter values. The privilege ID is issued on the VBusM read and write command bus such that the target endpoints can perform mMemory protection checks based on the PRIV of the external host that sets up the DMA transaction.	R	0x0

Bits	Field Name	Description	Type	Reset
7:4	RESERVED	Reserved	R Return 0's	0x0
3:0	PRIVID	Privilege ID SAMPPRXY.PRIVID is always updated with the value from configuration bus privilege ID field on any/every write to Program Set BIDX Register (trigger register). The PRIVID value for the SA Set and DF Set are copied from the value in the Program set along with the remainder of the parameter values. The privilege ID is issued on the VBusM read and write command bus such that the target endpoints can perform mMemory protection checks based on the privid of the external host that sets up the DMA transaction.	R	0x0

Table 16-400. Register Call Summary for Register EDMA_TPTCn_SAMPPRXY

Enhanced DMA
- EDMA Register Summary: [0] [1] [2] [3] [4] [5] [6] [7]

Table 16-401. EDMA_TPTCn_SACNTRLD

Address Offset	0x0000 0258		
Physical Address	0x4340 0258 0x4350 0258 0x40D0 5258 0x40D0 6258 0x4150 5258 0x4150 6258 0x01D0 5258 0x01D0 6258 0x4208 6258 0x4218 6258 0x4208 7258 0x4218 7258 0x4228 6258 0x4238 6258 0x4228 7258 0x4238 7258	**Instance**	SYS_EDMA_TPTC0 SYS_EDMA_TPTC1 DSP1_EDMA_TPTC0 DSP1_EDMA_TPTC1 DSP2_EDMA_TPTC0 DSP2_EDMA_TPTC1 DSP_EDMA_TPTC0 DSP_EDMA_TPTC1 EVE1_EDMA_TPTC0 EVE2_EDMA_TPTC0 EVE1_EDMA_TPTC1 EVE2_EDMA_TPTC1 EVE3_EDMA_TPTC0 EVE4_EDMA_TPTC0 EVE3_EDMA_TPTC1 EVE4_EDMA_TPTC1
Description	Source Active Set Count Reload		
Type	R		

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

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Reserved	R Return 0's	0x0
15:0	ACNTRLD	A-Cnt Reload value for Source Active Register set. Value copied from EDMA_TPTCn_PCNT[15:0] ACNT: Represents the originally programmed value of ACNT. The Reload value is used to reinitialize ACNT after each array is serviced (i.e., ACNT decrements to 0). by the Src offset in bytes between the starting address of each source array (recall that there are BCNT arrays of ACNT bytes)	R	0x0

Table 16-402. Register Call Summary for Register EDMA_TPTCn_SACNTRLD

Enhanced DMA
- EDMA Register Summary: [0] [1] [2] [3] [4] [5] [6] [7]

Table 16-403. EDMA_TPTCn_SASRCBREF

Address Offset	0x0000 025C		
Physical Address	0x4340 025C 0x4350 025C 0x40D0 525C 0x40D0 625C 0x4150 525C 0x4150 625C 0x01D0 525C 0x01D0 625C 0x4208 625C 0x4218 625C 0x4208 725C 0x4218 725C 0x4228 625C 0x4238 625C 0x4228 725C 0x4238 725C	**Instance**	SYS_EDMA_TPTC0 SYS_EDMA_TPTC1 DSP1_EDMA_TPTC0 DSP1_EDMA_TPTC1 DSP2_EDMA_TPTC0 DSP2_EDMA_TPTC1 DSP_EDMA_TPTC0 DSP_EDMA_TPTC1 EVE1_EDMA_TPTC0 EVE2_EDMA_TPTC0 EVE1_EDMA_TPTC1 EVE2_EDMA_TPTC1 EVE3_EDMA_TPTC0 EVE4_EDMA_TPTC0 EVE3_EDMA_TPTC1 EVE4_EDMA_TPTC1
Description	Source Active Set Source Address A-Reference		
Type	R		

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

Bits	Field Name	Description	Type	Reset
31:0	SADDRBREF	Source address reference for Source Active Register Set: Represents the starting address for the array currently being read. The next array's starting address is calculated as the 'reference address' plus the 'source b-idx' value.	R	0x0

Table 16-404. Register Call Summary for Register EDMA_TPTCn_SASRCBREF

Enhanced DMA

- EDMA Register Summary: [0] [1] [2] [3] [4] [5] [6] [7]

Table 16-405. EDMA_TPTCn_SADSTBREF

Address Offset	0x0000 0260		
Physical Address	0x4340 0260 0x4350 0260 0x40D0 5260 0x40D0 6260 0x4150 5260 0x4150 6260 0x01D0 5260 0x01D0 6260 0x4208 6260 0x4218 6260 0x4208 7260 0x4218 7260 0x4228 6260 0x4238 6260 0x4228 7260 0x4238 7260	**Instance**	SYS_EDMA_TPTC0 SYS_EDMA_TPTC1 DSP1_EDMA_TPTC0 DSP1_EDMA_TPTC1 DSP2_EDMA_TPTC0 DSP2_EDMA_TPTC1 DSP_EDMA_TPTC0 DSP_EDMA_TPTC1 EVE1_EDMA_TPTC0 EVE2_EDMA_TPTC0 EVE1_EDMA_TPTC1 EVE2_EDMA_TPTC1 EVE3_EDMA_TPTC0 EVE4_EDMA_TPTC0 EVE3_EDMA_TPTC1 EVE4_EDMA_TPTC1
Description	Source Active Destination Address B-Reference Register Reserved, return 0x0 w/o AERROR		
Type	R		

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

Bits	Field Name	Description	Type	Reset
31:0	DADDRBREF	Destination address reference is not applicable for Src Active Register Set. Reads return 0x0.	R	0x0

Table 16-406. Register Call Summary for Register EDMA_TPTCn_SADSTBREF

Enhanced DMA

- EDMA Register Summary: [0] [1] [2] [3] [4] [5] [6] [7]

Table 16-407. EDMA_TPTCn_DFCNTRLD

Address Offset	0x0000 0280		
Physical Address	0x4340 0280 0x4350 0280 0x40D0 5280 0x40D0 6280 0x4150 5280 0x4150 6280 0x01D0 5280 0x01D0 6280 0x4208 6280 0x4218 6280 0x4208 7280 0x4218 7280 0x4228 6280 0x4238 6280 0x4228 7280 0x4238 7280	**Instance**	SYS_EDMA_TPTC0 SYS_EDMA_TPTC1 DSP1_EDMA_TPTC0 DSP1_EDMA_TPTC1 DSP2_EDMA_TPTC0 DSP2_EDMA_TPTC1 DSP_EDMA_TPTC0 DSP_EDMA_TPTC1 EVE1_EDMA_TPTC0 EVE2_EDMA_TPTC0 EVE1_EDMA_TPTC1 EVE2_EDMA_TPTC1 EVE3_EDMA_TPTC0 EVE4_EDMA_TPTC0 EVE3_EDMA_TPTC1 EVE4_EDMA_TPTC1
Description	Destination FIFO Set Count Reload		
Type	R		

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

Bits	Field Name	Description	Type	Reset
31:16	RESERVED	Reserved	R Return 0's	0x0
15:0	ACNTRLD	A-Cnt Reload value for Destination FIFO Register set. Value copied from EDMA_TPTCn_PCNT[15:0] ACNT: Represents the originally programmed value of ACNT. The Reload value is used to reinitialize ACNT after each array is serviced (i.e., ACNT decrements to 0). by the Src offset in bytes between the starting address of each source array (recall that there are BCNT arrays of ACNT bytes)	R	0x0

Table 16-408. Register Call Summary for Register EDMA_TPTCn_DFCNTRLD

Enhanced DMA

- EDMA Register Summary: [0] [1] [2] [3] [4] [5] [6] [7]

Table 16-409. EDMA_TPTCn_DFSRCBREF

Address Offset	0x0000 0284		
Physical Address	0x4340 0284 0x4350 0284 0x40D0 5284 0x40D0 6284 0x4150 5284 0x4150 6284 0x01D0 5284 0x01D0 6284 0x4208 6284 0x4218 6284 0x4208 7284 0x4218 7284 0x4228 6284 0x4238 6284 0x4228 7284 0x4238 7284	**Instance**	SYS_EDMA_TPTC0 SYS_EDMA_TPTC1 DSP1_EDMA_TPTC0 DSP1_EDMA_TPTC1 DSP2_EDMA_TPTC0 DSP2_EDMA_TPTC1 DSP_EDMA_TPTC0 DSP_EDMA_TPTC1 EVE1_EDMA_TPTC0 EVE2_EDMA_TPTC0 EVE1_EDMA_TPTC1 EVE2_EDMA_TPTC1 EVE3_EDMA_TPTC0 EVE4_EDMA_TPTC0 EVE3_EDMA_TPTC1 EVE4_EDMA_TPTC1
Description	Destination FIFO Set Destination Address B Reference Reserved, return 0x0 w/o AERROR		
Type	R		

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

Bits	Field Name	Description	Type	Reset
31:0	SADDRBREF	Source address reference is not applicable for Dst FIFO Register Set. Reads return 0x0.	R	0x0

Table 16-410. Register Call Summary for Register EDMA_TPTCn_DFSRCBREF

Enhanced DMA

- EDMA Register Summary: [0] [1] [2] [3] [4] [5] [6] [7]

Table 16-411. EDMA_TPTCn_DFDSTBREF

Address Offset	0x0000 0288		
Physical Address	0x4340 0288 0x4350 0288 0x40D0 5288 0x40D0 6288 0x4150 5288 0x4150 6288 0x01D0 5288 0x01D0 6288	**Instance**	SYS_EDMA_TPTC0 SYS_EDMA_TPTC1 DSP1_EDMA_TPTC0 DSP1_EDMA_TPTC1 DSP2_EDMA_TPTC0 DSP2_EDMA_TPTC1 DSP_EDMA_TPTC0 DSP_EDMA_TPTC1
Description	Destination FIFO Set Destination Address A-Reference		
Type	R		

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

Bits	Field Name	Description	Type	Reset
31:0	DADDRBREF	Destination address reference for Dst FIFO Register Set: Represents the starting address for the array currently being written. The next array's starting address is calculated as the 'reference address' plus the 'dest bidx' value.	R	0x0

Table 16-412. Register Call Summary for Register EDMA_TPTCn_DFDSTBREF

Enhanced DMA

- EDMA Register Summary: [0] [1] [2] [3] [4] [5] [6] [7]

Table 16-413. EDMA_TPTCn_DFOPTi

Address Offset	0x0000 0300 + (0x40 * i)		
Physical Address	0x4340 0300 + (0x40 * i) 0x4350 0300 + (0x40 * i) 0x40D0 5300 + (0x40 * i) 0x40D0 6300 + (0x40 * i) 0x4150 5300 + (0x40 * i) 0x4150 6300 + (0x40 * i) 0x01D0 5300 + (0x40 * i) 0x01D0 6300 + (0x40 * i) 0x4208 6300 + (0x40 * i) 0x4218 6300 + (0x40 * i) 0x4208 7300 + (0x40 * i) 0x4218 7300 + (0x40 * i) 0x4228 6300 + (0x40 * i) 0x4238 6300 + (0x40 * i) 0x4228 7300 + (0x40 * i) 0x4238 7300 + (0x40 * i)	**Instance**	SYS_EDMA_TPTC0 SYS_EDMA_TPTC1 DSP1_EDMA_TPTC0 DSP1_EDMA_TPTC1 DSP2_EDMA_TPTC0 DSP2_EDMA_TPTC1 DSP_EDMA_TPTC0 DSP_EDMA_TPTC1 EVE1_EDMA_TPTC0 EVE2_EDMA_TPTC0 EVE1_EDMA_TPTC1 EVE2_EDMA_TPTC1 EVE3_EDMA_TPTC0 EVE4_EDMA_TPTC0 EVE3_EDMA_TPTC1 EVE4_EDMA_TPTC1
Description	Destination FIFO Set Options		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED									TCCHEN	RESERVED	TCINTEN	RESERVED		TCC						RESERVED	FWID			RESERVED	PRI			RESERVED		DAM	SAM

Bits	Field Name	Description	Type	Reset
31:23	RESERVED	Reserved	R Return 0's	0x0
22	TCCHEN	Transfer complete chaining enable 0x0: Transfer complete chaining is disabled. 0x1: Transfer complete chaining is enabled.	R	0x0
21	RESERVED	Reserved	R Return 0's	0x0
20	TCINTEN	Transfer complete interrupt enable 0x0: Transfer complete interrupt is disabled. 0x1: Transfer complete interrupt is enabled.	R	0x0
19:18	RESERVED	Reserved	R Return 0's	0x0
17:12	TCC	Transfer Complete Code The 6-bit code is used to set the relevant bit in CER or EDMA_TPCC_IPR of the TPCC module.	R	0x0
11	RESERVED	Reserved	R Return 0's	0x0
10:8	FWID	FIFO width control Applies if either SAM or DAM is set to FIFO mode.	R	0x0
7	RESERVED	Reserved	R Return 0's	0x0

Bits	Field Name	Description	Type	Reset
6:4	PRI	Transfer Priority 0x0: Priority 0 - Highest priority 0x1: Priority 1 ... 0x7: Priority 7 - Lowest priority	R	0x0
3:2	RESERVED	Reserved	R Return 0's	0x0
1	DAM	Destination Address Mode within an array 0x0: INCR, Dst addressing within an array increments. 0x1: FIFO, Dst addressing within an array wraps around upon reaching FIFO width.	R	0x0
0	SAM	Source Address Mode within an array 0x0: INCR, Source addressing within an array increments. 0x1: FIFO, Source addressing within an array wraps around upon reaching FIFO width.	R	0x0

Table 16-414. Register Call Summary for Register EDMA_TPTCn_DFOPTi

Enhanced DMA

- EDMA Register Summary: [0] [1] [2] [3] [4] [5] [6] [7]

Table 16-415. EDMA_TPTCn_DFSRCi

Address Offset	0x0000 0304 + (0x40 * i)		
Physical Address	0x4340 0304 + (0x40 * i) 0x4350 0304 + (0x40 * i) 0x40D0 5304 + (0x40 * i) 0x40D0 6304 + (0x40 * i) 0x4150 5304 + (0x40 * i) 0x4150 6304 + (0x40 * i) 0x01D0 5304 + (0x40 * i) 0x01D0 6304 + (0x40 * i) 0x4208 6304 + (0x40 * i) 0x4218 6304 + (0x40 * i) 0x4208 7304 + (0x40 * i) 0x4218 7304 + (0x40 * i) 0x4228 6304 + (0x40 * i) 0x4238 6304 + (0x40 * i) 0x4228 7304 + (0x40 * i) 0x4238 7304 + (0x40 * i)	**Instance**	SYS_EDMA_TPTC0 SYS_EDMA_TPTC1 DSP1_EDMA_TPTC0 DSP1_EDMA_TPTC1 DSP2_EDMA_TPTC0 DSP2_EDMA_TPTC1 DSP_EDMA_TPTC0 DSP_EDMA_TPTC1 EVE1_EDMA_TPTC0 EVE2_EDMA_TPTC0 EVE1_EDMA_TPTC1 EVE2_EDMA_TPTC1 EVE3_EDMA_TPTC0 EVE4_EDMA_TPTC0 EVE3_EDMA_TPTC1 EVE4_EDMA_TPTC1
Description	Destination FIFO source address register Reserved, return 0x0 w/o AERROR		
Type	R		

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

Bits	Field Name	Description	Type	Reset
31:0	SADDR	Source address is not applicable for Dst FIFO Register Set: Reads return 0x0.	R	0x0

Table 16-416. Register Call Summary for Register EDMA_TPTCn_DFSRCi

Enhanced DMA

- EDMA Register Summary: [0] [1] [2] [3] [4] [5] [6] [7]

Table 16-417. EDMA_TPTCn_DFCNTi

Address Offset	0x0000 0308 + (0x40 * i)		
Physical Address	0x4340 0308 + (0x40 * i) 0x4350 0308 + (0x40 * i) 0x40D0 5308 + (0x40 * i) 0x40D0 6308 + (0x40 * i) 0x4150 5308 + (0x40 * i) 0x4150 6308 + (0x40 * i) 0x01D0 5308 + (0x40 * i) 0x01D0 6308 + (0x40 * i) 0x4208 6308 + (0x40 * i) 0x4218 6308 + (0x40 * i) 0x4208 7308 + (0x40 * i) 0x4218 7308 + (0x40 * i) 0x4228 6308 + (0x40 * i) 0x4238 6308 + (0x40 * i) 0x4228 7308 + (0x40 * i) 0x4238 7308 + (0x40 * i)	**Instance**	SYS_EDMA_TPTC0 SYS_EDMA_TPTC1 DSP1_EDMA_TPTC0 DSP1_EDMA_TPTC1 DSP2_EDMA_TPTC0 DSP2_EDMA_TPTC1 DSP_EDMA_TPTC0 DSP_EDMA_TPTC1 EVE1_EDMA_TPTC0 EVE2_EDMA_TPTC0 EVE1_EDMA_TPTC1 EVE2_EDMA_TPTC1 EVE3_EDMA_TPTC0 EVE4_EDMA_TPTC0 EVE3_EDMA_TPTC1 EVE4_EDMA_TPTC1
Description	Destination FIFO count register		
Type	R		

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

Bits	Field Name	Description	Type	Reset
31:16	BCNT	B-Count Remaining for Dst Register Set: Number of arrays to be transferred, where each array is ACNT in length. Represents the amount of data remaining to be written. Initial value is copied from EDMA_TPTCn_PCNT. TC decrements ACNT and BCNT as necessary after each write dataphase is issued. Final value should be 0 when TR is complete.	R	0x0
15:0	ACNT	A-Count Remaining for Dst Register Set: Number of bytes to be transferred in first dimension. Represents the amount of data remaining to be written. Initial value is copied from EDMA_TPTCn_PCNT. TC decrements ACNT and BCNT as necessary after each write dataphase is issued. Final value should be 0 when TR is complete.	R	0x0

Table 16-418. Register Call Summary for Register EDMA_TPTCn_DFCNTi

Enhanced DMA

- EDMA Register Summary: [0] [1] [2] [3] [4] [5] [6] [7]

Table 16-419. EDMA_TPTCn_DFDSTi

Address Offset	0x0000 030C + (0x40 * i)		
Physical Address	0x4340 030C + (0x40 * i) 0x4350 030C + (0x40 * i) 0x40D0 530C + (0x40 * i) 0x40D0 630C + (0x40 * i) 0x4150 530C + (0x40 * i) 0x4150 630C + (0x40 * i) 0x01D0 530C + (0x40 * i) 0x01D0 630C + (0x40 * i) 0x4208 630C + (0x40 * i) 0x4218 630C + (0x40 * i) 0x4208 730C + (0x40 * i) 0x4218 730C + (0x40 * i) 0x4228 630C + (0x40 * i) 0x4238 630C + (0x40 * i) 0x4228 730C + (0x40 * i) 0x4238 730C + (0x40 * i)	**Instance**	SYS_EDMA_TPTC0 SYS_EDMA_TPTC1 DSP1_EDMA_TPTC0 DSP1_EDMA_TPTC1 DSP2_EDMA_TPTC0 DSP2_EDMA_TPTC1 DSP_EDMA_TPTC0 DSP_EDMA_TPTC1 EVE1_EDMA_TPTC0 EVE2_EDMA_TPTC0 EVE1_EDMA_TPTC1 EVE2_EDMA_TPTC1 EVE3_EDMA_TPTC0 EVE4_EDMA_TPTC0 EVE3_EDMA_TPTC1 EVE4_EDMA_TPTC1
Description	The destination FIFO destination address register		
Type	R		

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

Bits	Field Name	Description	Type	Reset
31:0	DADDR	Destination address for Dst FIFO Register Set: Initial value is copied from EDMA_TPTCn_PDST[31:0] DADDR. TC updates value according to destination addressing mode (EDMA_TPCC_OPT_n. SAM) and/or dest index value (BIDX. DBIDX) after each write command is issued. When a TR is complete, the final value should be the address of the last write command issued.	R	0x0

Table 16-420. Register Call Summary for Register EDMA_TPTCn_DFDSTi

Enhanced DMA

- EDMA Register Summary: [0] [1] [2] [3] [4] [5] [6] [7]

Table 16-421. EDMA_TPTCn_DFBIDXi

Address Offset	0x0000 0310 + (0x40 * i)		
Physical Address	0x4340 0310 + (0x40 * i) 0x4350 0310 + (0x40 * i) 0x40D0 5310 + (0x40 * i) 0x40D0 6310 + (0x40 * i) 0x4150 5310 + (0x40 * i) 0x4150 6310 + (0x40 * i) 0x01D0 5310 + (0x40 * i) 0x01D0 6310 + (0x40 * i) 0x4208 6310 + (0x40 * i) 0x4218 6310 + (0x40 * i) 0x4208 7310 + (0x40 * i) 0x4218 7310 + (0x40 * i) 0x4228 6310 + (0x40 * i) 0x4238 6310 + (0x40 * i) 0x4228 7310 + (0x40 * i) 0x4238 7310 + (0x40 * i)	**Instance**	SYS_EDMA_TPTC0 SYS_EDMA_TPTC1 DSP1_EDMA_TPTC0 DSP1_EDMA_TPTC1 DSP2_EDMA_TPTC0 DSP2_EDMA_TPTC1 DSP_EDMA_TPTC0 DSP_EDMA_TPTC1 EVE1_EDMA_TPTC0 EVE2_EDMA_TPTC0 EVE1_EDMA_TPTC1 EVE2_EDMA_TPTC1 EVE3_EDMA_TPTC0 EVE4_EDMA_TPTC0 EVE3_EDMA_TPTC1 EVE4_EDMA_TPTC1
Description	The destination FIFO B-index register		
Type	R		

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

Bits	Field Name	Description	Type	Reset
31:16	DBIDX	Dest B-Idx for Dest FIFO Register Set. Value copied from EDMA_TPTCn_PBIDX: B-Idx offset between Destination arrays: Represents the offset in bytes between the starting address of each destination array (recall that there are BCNT arrays of ACNT elements). DBIDX is always used, regardless of whether DAM is Increment or FIFO mode.	R	0x0
15:0	SBIDX	Dest B-Idx for Dest FIFO Register Set. Value copied from EDMA_TPTCn_PBIDX: B-Idx offset between Source arrays: Represents the offset in bytes between the starting address of each source array (recall that there are BCNT arrays of ACNT elements). SBIDX is always used, regardless of whether SAM is Increment or FIFO mode.	R	0x0

Table 16-422. Register Call Summary for Register EDMA_TPTCn_DFBIDXi

Enhanced DMA

- EDMA Register Summary: [0] [1] [2] [3] [4] [5] [6] [7]

Table 16-423. EDMA_TPTCn_DFMPPRXYi

Address Offset	0x0000 0314 + (0x40 * i)		
Physical Address	0x4340 0314 + (0x40 * i) 0x4350 0314 + (0x40 * i) 0x40D0 5314 + (0x40 * i) 0x40D0 6314 + (0x40 * i) 0x4150 5314 + (0x40 * i) 0x4150 6314 + (0x40 * i) 0x01D0 5314 + (0x40 * i) 0x01D0 6314 + (0x40 * i) 0x4208 6314 + (0x40 * i) 0x4218 6314 + (0x40 * i) 0x4208 7314 + (0x40 * i) 0x4218 7314 + (0x40 * i) 0x4228 6314 + (0x40 * i) 0x4238 6314 + (0x40 * i) 0x4228 7314 + (0x40 * i) 0x4238 7314 + (0x40 * i)	**Instance**	SYS_EDMA_TPTC0 SYS_EDMA_TPTC1 DSP1_EDMA_TPTC0 DSP1_EDMA_TPTC1 DSP2_EDMA_TPTC0 DSP2_EDMA_TPTC1 DSP_EDMA_TPTC0 DSP_EDMA_TPTC1 EVE1_EDMA_TPTC0 EVE2_EDMA_TPTC0 EVE1_EDMA_TPTC1 EVE2_EDMA_TPTC1 EVE3_EDMA_TPTC0 EVE4_EDMA_TPTC0 EVE3_EDMA_TPTC1 EVE4_EDMA_TPTC1
Description	The destination FIFO memory protection proxy register		
Type	R		

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

Bits	Field Name	Description	Type	Reset
31:9	RESERVED	Reserved	R Return 0's	0x0
8	PRIV	Privilege Level 0x0: User level privilege 0x1: Supervisor level privilege DFMPPRXY0.PRIV is always updated with the value from the configuration bus privilege field on any/every write to Program Set BIDX Register (trigger register). The PRIV value for the SA Set and DF Set are copied from the value in the Program set along with the remainder of the parameter values. The privilege ID is issued on the VBusM read and write command bus such that the target endpoints can perform mMemory protection checks based on the PRIV of the external host that sets up the DMA transaction.	R	0x0
7:4	RESERVED	Reserved	R Return 0's	0x0
3:0	PRIVID	Privilege ID: DFMPPRXY0.PRIVID is always updated with the value from configuration bus privilege ID field on any/every write to Program Set BIDX Register (trigger register). The PRIVID value for the SA Set and DF Set are copied from the value in the Program set along with the remainder of the parameter values. The privilege ID is issued on the VBusM read and write command bus such that the target endpoints can perform mMemory protection checks based on the privid of the external host that sets up the DMA transaction.	R	0x0

Table 16-424. Register Call Summary for Register EDMA_TPTCn_DFMPPRXYi

Enhanced DMA

- EDMA Register Summary: [0] [1] [2] [3] [4] [5] [6] [7]