Application Report
SPRAAA6–December 2005

Migrating from EDMA v2.0 to EDMA v3.0 for TMS320DM644X DMSoC

Xiaohui Li *DSP Software Applications*

1 Introduction

A newly designed enhanced direct memory access (EDMA3) is introduced on the TMS320DM644x Digital Media System-on-Chip (DMSoC). The EDMA3 has many new features that improve system performance and enhance debugging capabilities. This document summarizes the key differences between the EDMA3 used on DM644x devices and the EDMA2 used on TMS320C64x devices and provides guidance for migrating from EDMA2 to EDMA3. For more details about EDMA3, see the *TMS320DM644x DMSoC Enhanced Direct Memory Access (EDMA3) Controller User's Guide* (SPRUE23). For details about EDMA2, see the *TMS3206000 DSP Enhanced Direct Memory Access (EDMA) Controller Reference Guide* (SPRU234).

1.1 System Architecture Overview

Figure 1 shows the high-level system architecture of the DM644x device.

As shown in Figure 1, the DM644x device consists of masters, slaves, and a switched central resource (SCR). The masters and slaves are cross connected through the SCR. Masters initiate data transfer between masters and slaves. For example, the masters include the ARM processor, C64x+ CPU (DSP), video port subsystem (VPSS), EDMA transfer controllers, and a crossbar port through which six master peripherals including USB, EMAC, ATA/CF, and VLYNQ are connected to the SCR. The slaves include the ARM memory, C64x+ DSP memory, DDR EMIF, and a crossbar through which many peripherals are connected to the SCR.

The SCR provides connectivity and arbitrations among masters and slaves. It allows truly concurrent data transfers between unique connections between masters and slaves. For example, the connection between USB to DDR2 EMIF is independent of the connection between ASP and DSP L2. The two data transfers are completely in parallel. SCR handles arbitration when multiple masters access the same slave. The priority levels among the masters are user programmable.

On C64x devices, both regular EDMA transfers and transfers between master peripherals and slaves all go through the EDMA2 transfer controller. So all the EDMA transfers and other master peripheral transfers compete for the bandwidth sources of EDMA2 transfer controller.

There is a key difference between the DM644x device-based system architecture and the C64x DSP-based system architecture. On DM644x devices, the EDMA3 transfer controller and other master peripherals such as ATA/CF and USB on the DM644x device are all masters. EDMA3 only handles the data transfer between slaves. The transfers between other master peripherals and slaves do not go through EDMA.

In summary, the system architecture of DM644x devices provides more concurrence and offers increased system performances

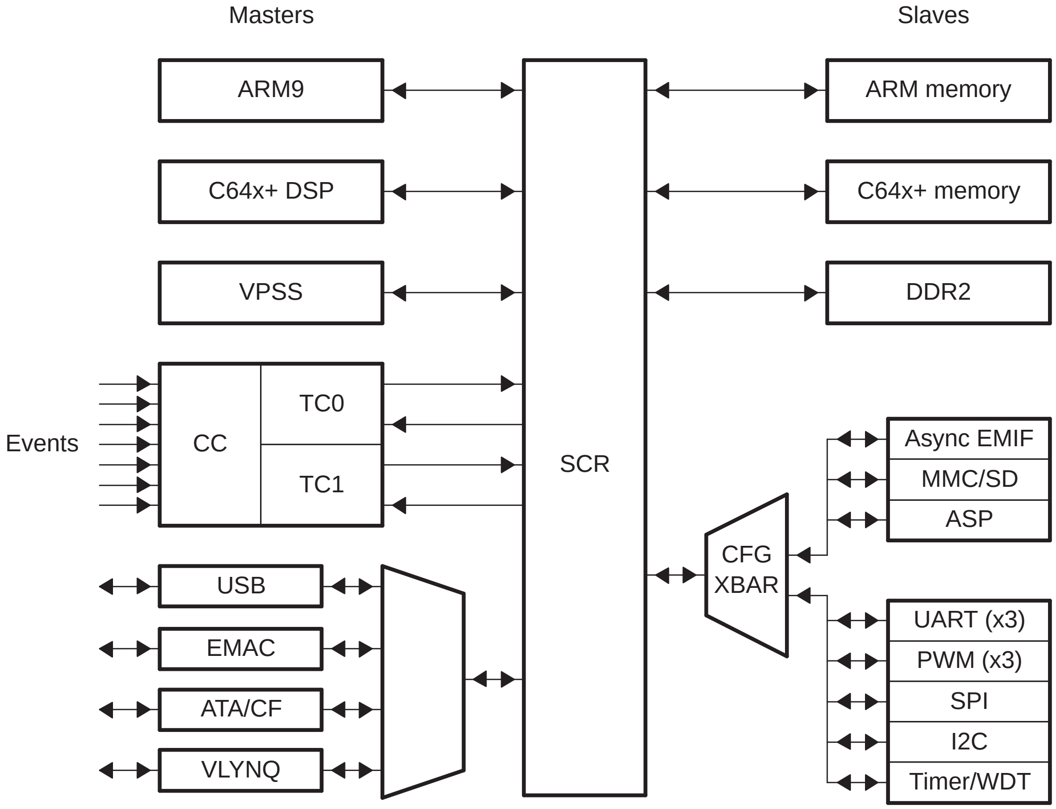


Figure 1. TMS320DM644x Device Block Diagram

1.2 EDMA3 Overview

The EDMA3 consists of a channel controller (CC) and two transfer controllers (TC), see Figure 2. You can program EDMA3 for data transfer between two slaves of a switched central resource (SCR). Programmed transfers are typically between a slave peripheral and a memory (for example, ASP to/from L2 SRAM) or between two slave memories (for example, L2 SRAM to/from EMIF). The EDMA3 supports 64 DMA channels and 8 QDMA channels.

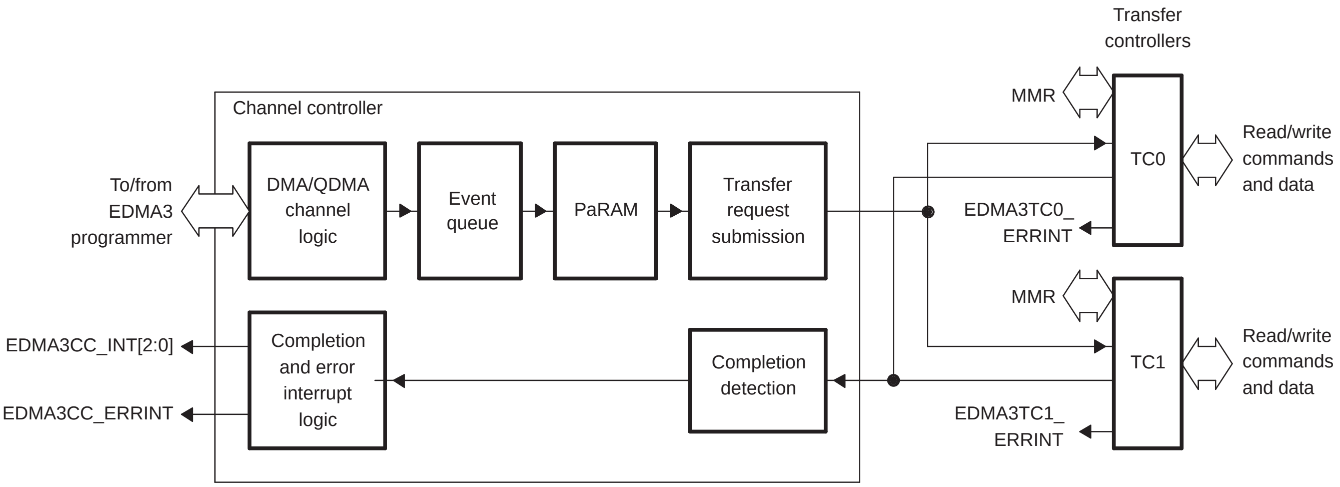


Figure 2. EDMA3 Block Diagram

1.2.1 EDMA3 Channel Controller Overview

Figure 3 shows a high-level block diagram for the EDMA3 channel controller (CC). The channel controller serves as the user interface for EDMA3. It provides functionalities of detecting trigger events, prioritization of DMA events, queuing, and submitting transfer requests to the EDMA transfer controller (TC) for data movement between slave peripherals.

EDMA3 supports two types of DMA channels, regular DMA channels and QDMA channels. DMA channels can be triggered by external events (for example, ASP transmit event and receive event), by software writing a 1 to a given bit location of the event set register of a channel, or by way of chaining. QDMA channels can be triggered by writing to a designated QDMA trigger word.

The CC arbitrates among all pending DMA/QDMA events with a fixed 64:1 and 8:1 priority encoder for DMA and QDMA events, respectively (the lowest channel number is the highest priority). DMA events always have higher priority than QDMA events. The winning event is queued into the event queue. There are two event queues. Each event queue is serviced in FIFO order, with a maximum of 16 queued events per event queue. If more than one transfer controller is ready to be programmed with a transfer request, the event queues are serviced with fixed priority, Q0 is always higher than Q1.

The CC contains 128 parameter RAM (PaRAM) that supports flexible ping-pong, circular buffering, channel chaining, auto reloading, memory protection etc.

Events are extracted from the event queue as soon as the transfer controller is available for a new transfer request to be programmed into the transfer controller. As an event is extracted from the event queue, the associated PaRAM entry is processed and submitted to the transfer controller. The CC updates the appropriate counts and addresses in the PaRAM entry in anticipation of the next trigger event for that PaRAM entry.

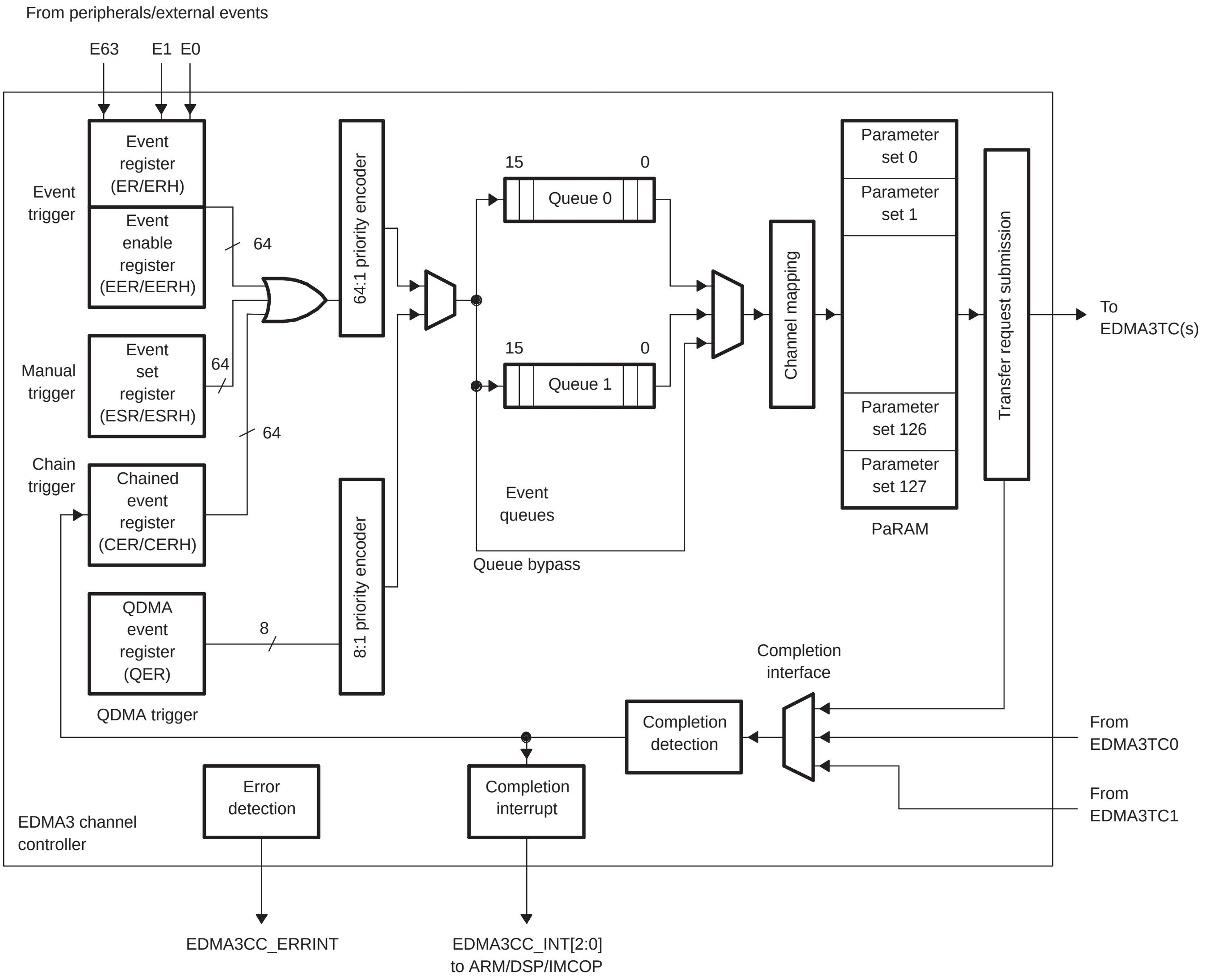


Figure 3. EDMA3 Channel Controller Block Diagram

1.2.2 EDMA3 Transfer Controller Overview

The transfer controller’s primary responsibility is to perform read and write transfers to the slaves connected to SCR as specified by the PaRAM entries. Figure 4 shows a high-level block diagram of the EDMA3 transfer controller. For more details, see the *TMS320DM644x DMSoC Enhanced Direct Memory Access (EDMA3) Controller User's Guide* (SPRUE23).

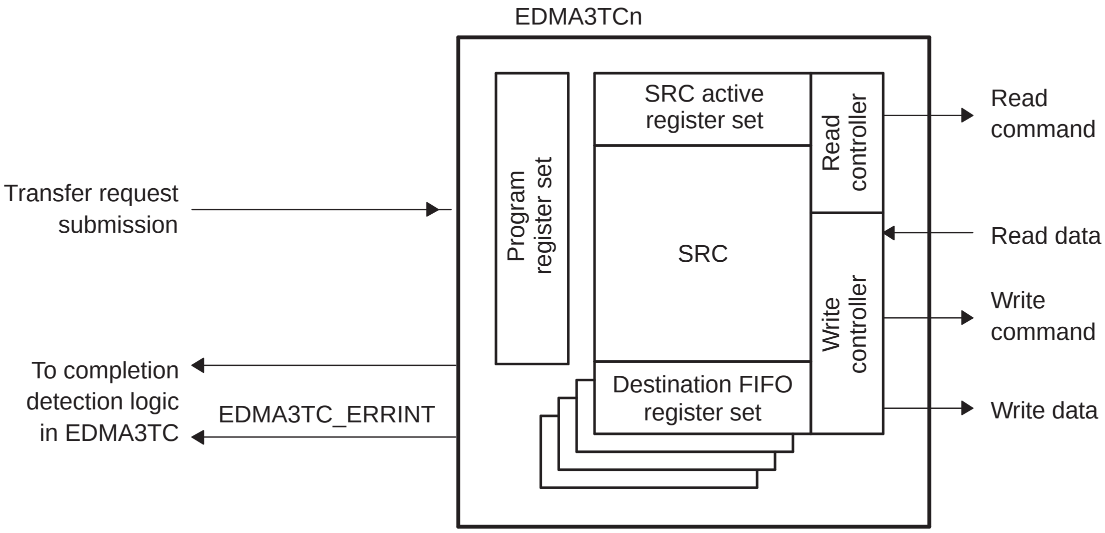


Figure 4. EDMA3 Transfer Controller Block Diagram

1.3 Architecture Comparison Between EDMA3 and EDMA2

On EDMA3-based devices, both the EDMA channel controllers and other Master peripherals are all Masters of a switched central resource (SCR). The EDMA channel controllers and transfer controllers only handle data transfers between two Slaves, such as ASP to DDR2 and/or DDR2 to DSP L2 memory. All other Master peripherals are directly connected to SCR. There are two levels of prioritization related with EDMA.

- One level is within the EDMA channel controller that handles prioritization and arbitration among all the DMA and QDMA schemes.
- The second level is the system level prioritization that configures the prioritization of the EDMA transfer controller versus other Masters. This system level priority resolution is performed by the SCR.

On EDMA2, all the DMA and master transfers go through the EDMA transfer controller. All the DMA transfers and other master transfers compete for the transfer controller bandwidth. The prioritization among all the transfers is done within the transfer controller.

In comparison with EDMA2, the EDMA3-based devices provide more bandwidth and concurrence for both DMA/QDMA and other master related data transfers. It is capable of providing concurrent transfers for both DMA/QDMA and master transfers, assuming all the masters go to different slaves.

1.4 Overview Feature Differences and New Features on EDMA3

Table 1 summarizes the major differences between EDMA3 and EDMA2.

Table 1. Summary of Major Differences Between EDMA3 and EDMA2

Feature	Description	EDMA2	EDMA3
PaRAM	PaRAM entry	6 32-bit words	8 32-bit words
	Number of PaRAM entries	Number depends on devices	128
	Event mapping to PaRAM	Fixed	Fixed
	Indexing	Same for source and destination	Different for source and destination
Transfer Type	Dimension of transfer	Two	Three
	Synchronization types	Element synchronized 1D	A-synchronized transfer
		Frame synchronized 1D	AB-synchronized transfer
		Array synchronized 2D	
		Block synchronized 2D	
QDMA	QDMA location	Outside of EDMA	Part of EDMA
	QDMA channel mapping	NA	Flexible
	QDMA triggering	Write to QDMA pseudo register	Writing to trigger word
	QDMA linking	NA	Flexible
	STATIC bit in OPT for QDMA	NA	New feature on EDMA3
Interrupt	Interrupt types	No error interrupt	Transfer completion interrupt error interrupt
	Interrupt registers		Some new interrupt registers are introduced
	Region interrupt	NA	New feature on EDMA3
Event Queue	Number of event queues	4	2
	Queue length	16	16
	Queue utilization	Shared by DMA/QDMA and other master peripherals	Used only by EDMA/QDMA channels
	Event queue timing	Fire-and-Forget	Need to wait for TR submitted to TC to update PaRAM
Priority Processing	Mapping between events and event queues	Through PRI of OPT field	Through queue number registers
	Queue priority versus other master transfers	Arbitrated at transfer controller (TC)	Arbitrated at switch center resource (SCR)
Linking	DMA linking feature availability	Available, optional	Always
	QDMA linking feature availability	NA	Always
	Terminating linking	Link to a NULL entry	Configure FFFFh of LINK field of last linked PaRAM
Chaining	Chaining comparison	Use same bits for enabling chain and interrupt	Use different bits for enabling chain and interrupt
Set and Clear	Set and clear operation	NA	New feature on EDMA3
DMA Region Access	DMA region access	NA	New feature on EDMA3
Debug Visibility	Event queue visibility	NA	New feature on EDMA3
	Event queue watermarking	NA	New feature on EDMA3
	Event queue threshold	NA	New feature on EDMA3
Error Detection	Event missed registers	NA	New feature on EDMA3
	Channel controller error registers	NA	New feature on EDMA3
	Error from null parameter entry	NA	New feature on EDMA3
	Error interrupts	NA	New feature on EDMA3

2 Comparison Between EDMA3 and EDMA2

2.1 PaRAM Set

Similar to EDMA2, the EDMA3 controller is also a RAM-based architecture and each EDMA channel transfer context (source/destination addresses, count, indexes, etc.) is programmed in a parameter table (see Figure 5 and Figure 6). The table is a block of parameter RAM (PaRAM) located within the EDMA channel controller. Although both EDMA3 and EDMA2 use RAM-based architecture, there are key differences related to PaRAM entries as summarized in this section.

	31	0
Word 0	Channel Options Parameter (OPT)	
Word 1	Channel Source Address (SRC)	
Word 2	Count for 2nd Dimension (BCNT)	Count for 1st Dimension (ACNT)
Word 3	Channel Destination Address (DST)	
Word 4	Destination BCNT Index (DSTBIDX)	Source BCNT Index (SRCBIDX)
Word 5	BCNT Reload (BCNTRLD)	Link Address (LINK)
Word 6	Destination CCNT Index (DSTCIDX)	Source CCNT Index (SRCCIDX)
Word 7	Reserved	Count for 3rd Dimension (CCNT)

Figure 5. PaRAM Entry for EDMA3

	31	0
Word 0	Channel Options Parameter (OPT)	
Word 1	Channel Source Address (SRC)	
Word 2	Array/Frame Count (FRMCNT)	Element Count (ELECNT)
Word 3	Channel Destination Address (DST)	
Word 4	Array/Frame Index (FRMIDX)	Element Index (ELEIDX)
Word 5	Element Count Reload (ELERLD)	Link Address (LINK)

Figure 6. PaRAM Entry for EDMA2

2.1.1 PaRAM Entries

On EDMA3 (Figure 5), the PaRAM entry size is eight 32-bit words; on EDMA2 (Figure 6), the PaRAM entry size is six 32-bit words. Table 2 summarizes the comparisons between the two PaRAM entries.

Table 2. Summary of PaRAM Comparison Between EDMA3 and EDMA2

PaRAM Entry	EDMA2	EDMA3
OPT field	See the *TMS320C6000 DSP Enhanced Direct Memory Access (EDMA) Controller Reference Guide* (SPRU234).	Significant changes have been introduced. See the *TMS320DM644x DMSoC Enhanced Direct Memory Access (EDMA3) Controller User's Guide* (SPRUE23).
Count	Only ELECNT and FRMCNT are available to specify 2-dimension transfers.	ACNT, BCNT, and CCNT are introduced to specify the sizes for 3-dimension transfers.
Index	Both source and destination use the same indexes ELEIDX and FRMIDX.	Source indexes (SRCBIDX and SRCCIDX) and destination indexes (DSTBIDX and DSTCIDX) are introduced and allow independent source and destination indexing.
DST and SRC fields	Both EDMA3 and EDMA2 have 32-bit source and destination fields specifying the source and destination addresses.	
Link field	Both EDMA3 and EDMA2 have 16-bit link field specifying the linking address.	
Reload field	The ELERLD field on EDMA2 is equivalent to the BCNTRLD field on EDMA3	

2.1.2 Number of PaRAM Entries

Table 3 summarizes the number of PaRAM entries for some EDMA3-based devices and EDMA2-based devices. See your device-specific data manual for the number of PaRAM entries on your device.

Table 3. Number of PaRAM Entries for Different Devices

Device	EDMA Type	Number of PaRAM Entries
TMS320C6415 DSP	EDMA2	83
TMS320DM642 DSP	EDMA2	200
TMS320DM644x DMSoC	EDMA3	128

2.1.3 Event Mapping to PaRAM

On DM644x devices, the mapping between the DMA numbers and the PaRAM sets is a fixed, one-to-one mapping. For example, channel (event) 0 is mapped to PaRAM set 0, channel (event) 1 is mapped to PaRAM set 1, etc. The mapping between the QDMA channels and the PaRAM sets is programmable. Any of the 8 QDMA channels can be mapped to any of the 128 PaRAM entries through QDMA channel mapping registers QCHMAP0 to QCHMAP7 shown in Figure 7.

Similarly on EDMA2, the mapping between the 64 DMA channels and the PaRAM entries and the mapping between the DMA events and the PaRAM entries are fixed. For example, event 0 is permanently mapped to the first PaRAM entry and event 1 is permanently mapped to the second PaRAM. Five QDMA registers, not PaRAM, are used for configuration of QDMA transfers.

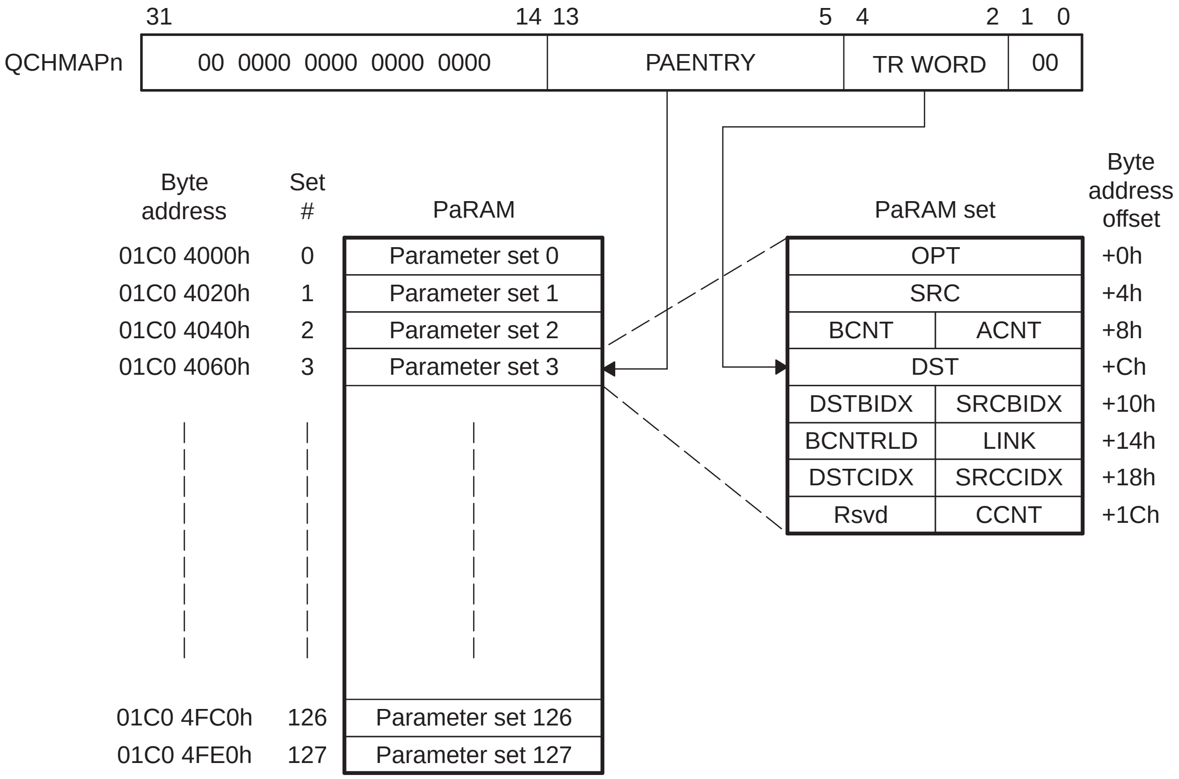


Figure 7. DMA and QDMA Channel Mapping to PaRAM

2.1.4 Dimension of Transfer

Unlike EDMA2, the EDMA3 uses more orthogonal structure for its DMA transfers. Each transfer is always defined by 3 dimensions. Of the 3 dimensions, only two synchronization types are supported: A-synchronized transfers and AB-synchronized transfers. (ABC-synchronized transfers can be logically achieved using chaining). The 3 dimensions are shown in Figure 8 and summarized as follows:

- 1st Dimension or Array (A): The 1st dimension in a transfer consists of ACNT contiguous bytes. Total transfer size is ACNT bytes.
- 2nd Dimension or Frame (B): The 2nd dimension in a transfer consists of BCNT arrays of ACNT bytes. Each array transfer in the 2nd dimension is separated from each other by an index programmed using SRCBIDX or DSTBIDX. Total transfer size is ACNT × BCNT bytes.
- 3rd Dimension or Block (C): The 3rd dimension in a transfer consists of CCNT frames of BCNT arrays programmed using SRCCIDX or DSTCIDX. Total transfer size is ACNT × BCNT × CCNT bytes.

On EDMA2, the transfer is defined as:

- 1st Dimension or Frame: The 1st dimension in a transfer consists of ELECNT elements and the consecutive elements are separated by ELEIDX. The element size is defined as ELECNT × ESIZE bytes, if ESIZE is configured as number of bytes.
- 2nd Dimension: The 2nd dimension in a transfer consists of FRMCNT frames of ELECNT elements. Each frame transfer in the 2nd dimension is separated from each other by FRMIDX index. Total transfer size is FRMCNT × ELECNT × ESIZE bytes, if ESIZE is configured as number of bytes.

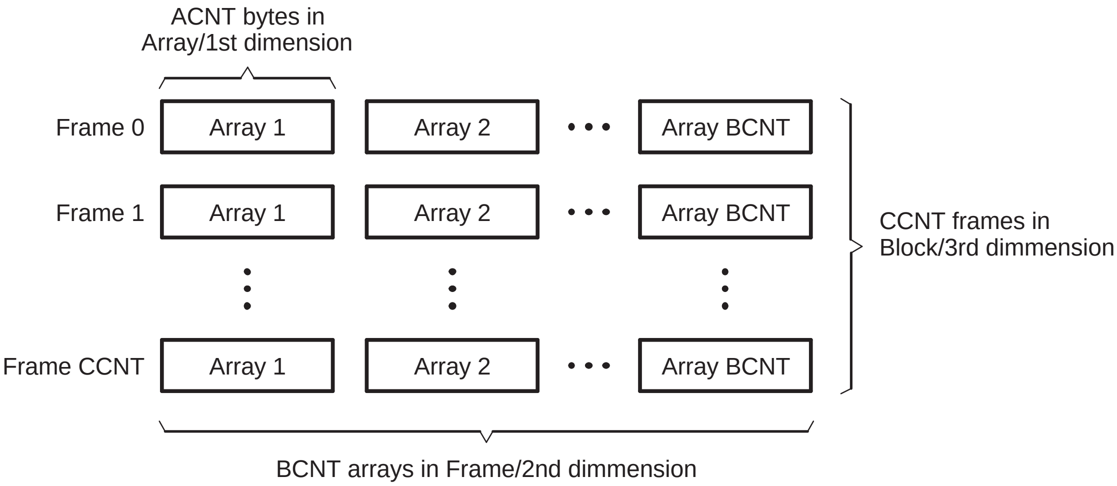


Figure 8. Dimension of Array, Frame, and Block

2.1.5 Indexing

EDMA2 uses the same two indexes, ELEIDX and FRMIDX, to specify the offset between the consecutive two elements or two frames for both source and destination.

EDMA3 uses different indexes for source and destinations. It uses SRCBIDX and DSTBIDX to specify the offsets between the two consecutive arrays and two consecutive frames for source indexing. It uses DSTBIDX and DSTCIDX to specify destination indexing. This means that on EDMA3, the source and destination can use completely different indexing. Figure 9 and Figure 10 show two examples demonstrating the potential indexing differences between EDMA2 and EDMA3.

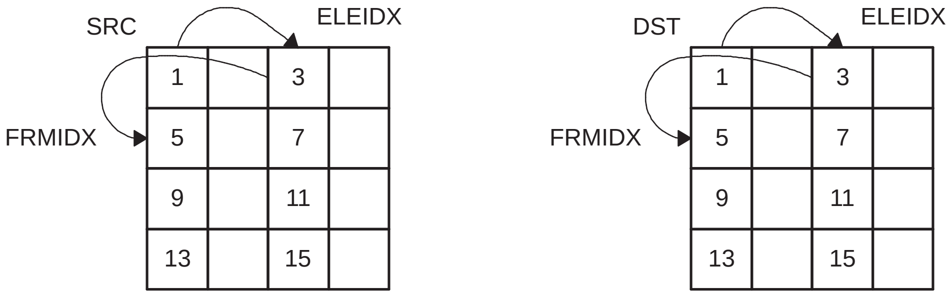


Figure 9. Indexing Example for EDMA2

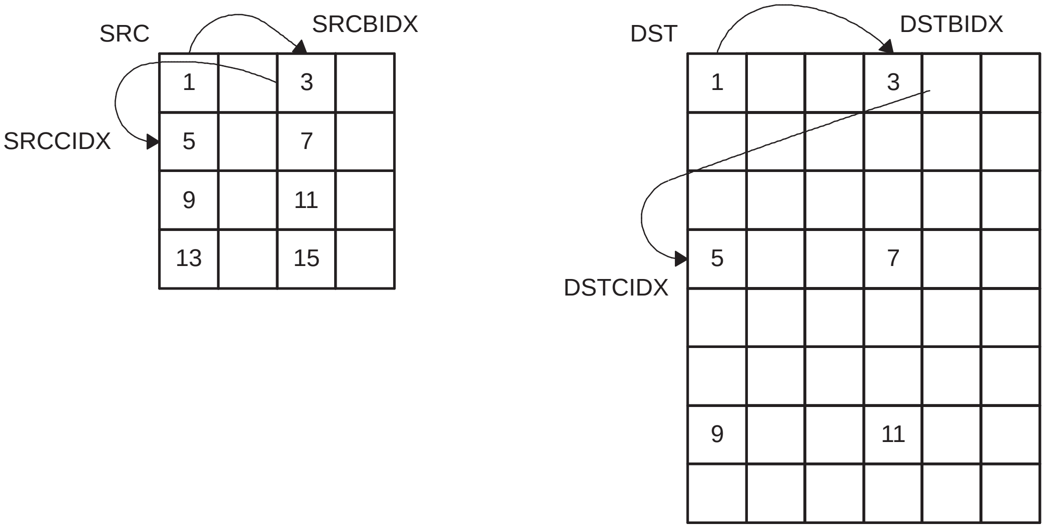


Figure 10. Indexing Example for EDMA3

2.2 *EDMA3 Synchronization Type*

There are two synchronization types, A and AB, on EDMA3:

- *A-Synchronized Transfers:* In an A-synchronized transfer, each transfer request sync triggers the transfer of the 1st dimension of ACNT bytes, or one array of ACNT bytes. In other words, each TR packet consists of transfer information for one array only. Arrays can be separated by SRCBIDX and DSTBIDX as shown in Figure 11, where the start address of Array N is equal to the start address of Array N – 1 + (SRCBIDX or DSTBIDX).
- *AB-Synchronized Transfers*: In an AB-synchronized transfer, each transfer request sync triggers the transfer for 2 dimensions or for one entire frame of BCNT of ACNT bytes. Arrays can be separated by SRCBIDX and DSTBIDX and frames can be separated by SRCIDX and DSTIDX. After a transfer request is submitted, the addresses are updated by adding SRCIDX and DSTIDX to the beginning addresses of the beginning array in the frame as shown in Figure 12.

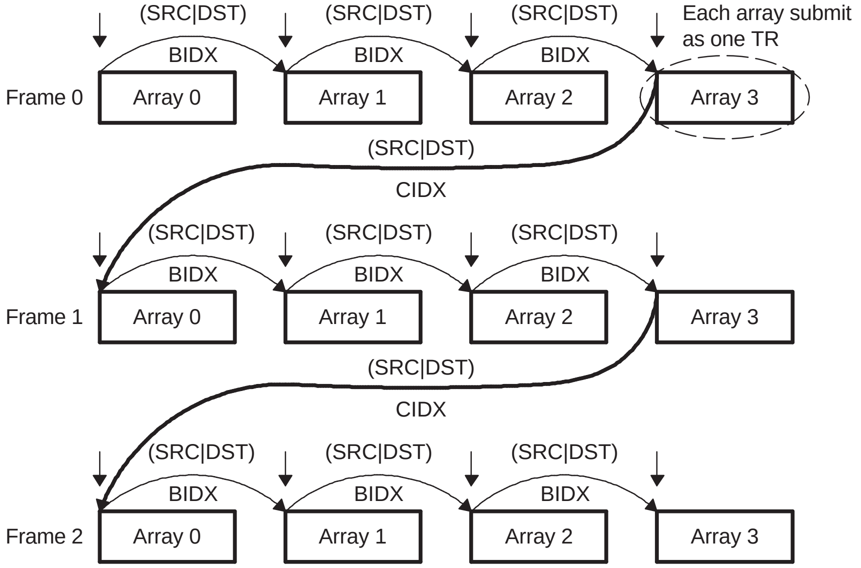


Figure 11. A-Synchronized Transfers (ACNT = n, BCNT = 4, CCNT = 3)

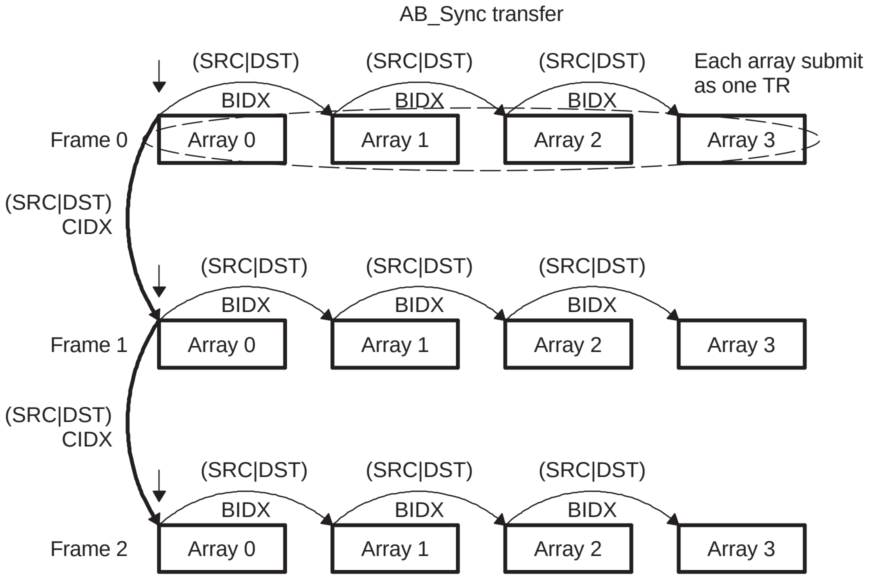


Figure 12. AB-Synchronized Transfers (ACNT = n, BCNT = 4, CCNT = 3)

EDMA2 has the following synchronization types for its DMA transfers:

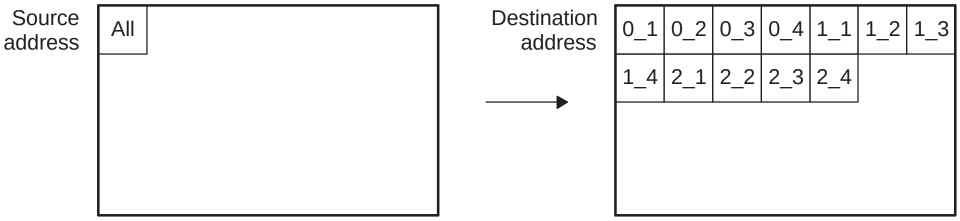
- Element Synchronized 1D Transfer (FS = 0)
- Frame Synchronized 1D Transfer (FS = 1)
- Array Synchronized 2D Transfer (FS = 0)
- Block Synchronized 2D Transfer (FS = 1)

For more details on EDMA2 synchronization types, see the *TMS320C6000 DSP Enhanced Direct Memory Access (EDMA) Controller Reference Guide* (SPRU234).

2.3 Example of Element Synchronized 1D-to-1D Transfer on EDMA2

Figure 13 shows a block diagram of an element synchronized 1D-to-1D transfer and the configuration on EDMA2. The source address is from a fixed location. The data is copied to continues addresses in destination.

(a) Transfer Path



(b) EDMA Parameters

Parameter Contents	
2020 0000h	
Source Address	
0002h	0004h
Destination Address	
Don't care	Don't care
0004h	Don't care

Parameter	
Channel Options Parameter (OPT)	
Channel Source Address (SRC)	
Array/Frame Count (FRMCNT)	Element Count (ELECNT)
Channel Destination Address (DST)	
Array/Frame Index (FRMIDX)	Element Index (ELEIDX)
Element Count Reload (ELERLD)	Link Address (LINK)

(c) EDMA Channel Options Parameter (OPT) Content

31 29	28 27	26	25 24	23	22 21	20	19 16
001	00	0	00	0	01	0	0000
PRI	ESIZE	2DS	SUM	2DD	DUM	TCINT	TCC

15	14 13	12 2	1	0
0	00	000 0000 0000	0	0
Rsvd	TCCM	Reserved	LINK	FS

Figure 13. Element Synchronized 1D-to-1D Transfer (SUM = 00, DUM = 01) on EDMA2

On EDMA3, the A-synchronization type is used for this EDMA2 element synchronized 1D-to-1D transfer. At every synchronization event, ACNT bytes are transferred from source to destination until all the counters are exhausted. Figure 14 shows the configuration for PaRAM entry and the channel options parameter (OPT) content for this transfer example.

(a) EDMA Parameters

Parameter Contents	
0010 000Bh	
Source Address	
0004h	0004h
Destination Address	
0004h	0004h
0004h	FFFFh
0004h	0004h
0000h	0003h

Parameter	
Channel Options Parameter (OPT)	
Channel Source Address (SRC)	
Count for 2nd Dimension (BCNT)	Count for 1st Dimension (ACNT)
Channel Destination Address (DST)	
Destination BCNT Index (DSTBIDX)	Source BCNT Index (SRCBIDX)
BCNT Reload (BCNTRLD)	Link Address (LINK)
Destination CCNT Index (DSTCIDX)	Source CCNT Index (SRCCIDX)
Reserved	Count for 3rd Dimension (CCNT)

(b) EDMA Channel Options Parameter (OPT) Content

31	30 28	27 24	23	22	21	20	19 18	17 16
0	000	0000	0	0	0	1	xx	00
PRIV	Reserved	PRIVID	ITCCHEN	TCCHEN	ITCINTEN	TCINTEN	Reserved	TCC

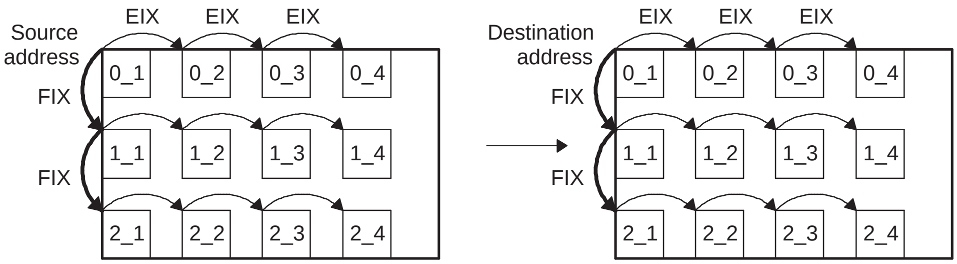
15 12	11	10 8	7 4	3	2	1	0
0000	0	000	0000	1	0	1	1
TCC	TCCMOD	FWID	Reserved	STATIC	SYNCDIM	DAM	SAM

Figure 14. Element Synchronized 1D-to-1D Transfer on EDMA3

2.4 Example of Frame Synchronized 1D-to-1D Transfer on EDMA2 and EDMA3

Figure 15 shows a block diagram of a frame synchronized 1D-to-1D transfer and the configuration on EDMA2. At every synchronization event, a frame of elements is transferred from source location to destination.

(a) Transfer Path



(b) EDMA Parameters

Parameter Contents	
2360 0001h	
Source Address	
0002h	0004h
Destination Address	
FIX (frame index)	EIX (element index)
Don't care	Don't care

Parameter	
Channel Options Parameter (OPT)	
Channel Source Address (SRC)	
Array/Frame Count (FRMCNT)	Element Count (ELECNT)
Channel Destination Address (DST)	
Array/Frame Index (FRMIDX)	Element Index (ELEIDX)
Element Count Reload (ELERLD)	Link Address (LINK)

(c) EDMA Channel Options Parameter (OPT) Content

31 29	28 27	26	25 24	23	22 21	20	19 16
001	00	0	11	0	11	0	0000
PRI	ESIZE	2DS	SUM	2DD	DUM	TCINT	TCC

15	14 13	12 2	1	0
0	00	000 0000 0000	0	1
Rsvd	TCCM	Reserved	LINK	FS

Figure 15. Frame Synchronized 1D-to-1D Transfer (SUM = 11, DUM = 11) on EDMA2

On EDMA3, the AB-synchronization type is used for this EDMA2 frame synchronized 1D-to-1D transfer. At every synchronization event, ACNT × BCNT bytes are transferred from source to destination until all the counters are exhausted. Figure 16 shows the configuration for PaRAM entry and the channel options parameter (OPT) content for this transfer example.

(a) EDMA Parameters

Parameter Contents	
0010 000Fh	
Source Address	
0004h	0004h
Destination Address	
EIX (element index)	EIX (element index)
0000h	FFFFh
FIX (frame index)	FIX (frame index)
0000h	0003h

Parameter	
Channel Options Parameter (OPT)	
Channel Source Address (SRC)	
Count for 2nd Dimension (BCNT)	Count for 1st Dimension (ACNT)
Channel Destination Address (DST)	
Destination BCNT Index (DSTBIDX)	Source BCNT Index (SRCBIDX)
BCNT Reload (BCNTRLD)	Link Address (LINK)
Destination CCNT Index (DSTCIDX)	Source CCNT Index (SRCCIDX)
Reserved	Count for 3rd Dimension (CCNT)

(b) EDMA Channel Options Parameter (OPT) Content

31	30 28	27 24	23	22	21	20	19 18	17 16
0	000	0000	0	0	0	1	xx	00
PRIV	Reserved	PRIVID	ITCCHEN	TCCHEN	ITCINTEN	TCINTEN	Reserved	TCC

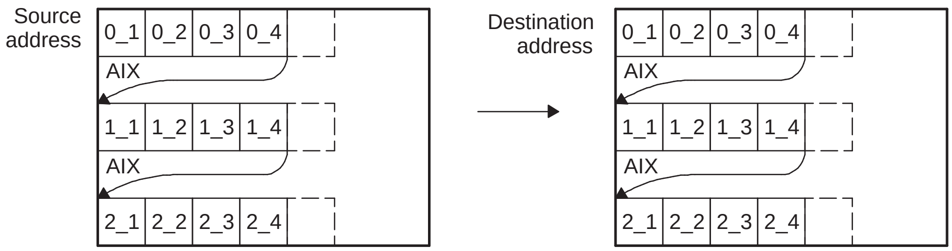
15 12	11	10 8	7 4	3	2	1	0
0000	0	000	0000	1	1	1	1
TCC	TCCMOD	FWID	Reserved	STATIC	SYNCDIM	DAM	SAM

Figure 16. Frame Synchronized 1D-to-1D Transfer on EDMA3

2.5 Example of Block Synchronized 2D-to-2D Transfer on EDMA2 and EDMA3

Figure 17 shows a block diagram of a block synchronized 2D-to-2D transfer and the configuration on EDMA2. At every synchronization event, a block of several frames of data is copied from source address to destination address.

(a) Transfer Path



(b) EDMA Parameters

Parameter Contents	
65A0 0001h	
Source Address	
0002h	0004h
Destination Address	
AIX (array index)	Don't care
Don't care	Don't care

Parameter	
Channel Options Parameter (OPT)	
Channel Source Address (SRC)	
Array/Frame Count (FRMCNT)	Element Count (ELECNT)
Channel Destination Address (DST)	
Array/Frame Index (FRMIDX)	Element Index (ELEIDX)
Element Count Reload (ELERLD)	Link Address (LINK)

(c) EDMA Channel Options Parameter (OPT) Content

31 29	28 27	26	25 24	23	22 21	20	19 16
011	00	1	01	1	01	0	0000
PRI	ESIZE	2DS	SUM	2DD	DUM	TCINT	TCC

15	14 13	12 2	1	0
0	00	000 0000 0000	0	1
Rsvd	TCCM	Reserved	LINK	FS

Figure 17. Block Synchronized 2D-to-2D Transfer (SUM = 01, DUM = 01) on EDMA2

On EDMA3, the AB-synchronization type is used for this EDMA2 block synchronized 2D-to-2D transfer. At every synchronization event, a block of data specified by ACNT × BCNT × CCNT bytes is transferred from source to destination. Figure 18 shows the configuration for PaRAM entry and the channel options parameter (OPT) content for this transfer example.

(a) EDMA Parameters

Parameter Contents	
0010 000Fh	
Source Address	
0003h	0010h
Destination Address	
10h + AIX (array index)	10h + AIX (array index)
0000h	FFFFh
0000h	0000h
0000h	0001h

Parameter	
Channel Options Parameter (OPT)	
Channel Source Address (SRC)	
Count for 2nd Dimension (BCNT)	Count for 1st Dimension (ACNT)
Channel Destination Address (DST)	
Destination BCNT Index (DSTBIDX)	Source BCNT Index (SRCBIDX)
BCNT Reload (BCNTRLD)	Link Address (LINK)
Destination CCNT Index (DSTCIDX)	Source CCNT Index (SRCCIDX)
Reserved	Count for 3rd Dimension (CCNT)

(b) EDMA Channel Options Parameter (OPT) Content

31	30 28	27 24	23	22	21	20	19 18	17 16
0	000	0000	0	0	0	1	xx	00
PRIV	Reserved	PRIVID	ITCCHEN	TCCHEN	ITCINTEN	TCINTEN	Reserved	TCC

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

Figure 18. Block Synchronized 2D-to-2D Transfer on EDMA3

2.6 QDMA Comparison Between EDMA3 and EDMA2

2.6.1 QDMA Architecture Differences Between EDMA3 and EDMA2

Table 4 summarizes the QDMA architecture differences between EDMA3 and EDMA2.

Table 4. Summary of QDMA Architecture Differences Between EDMA3 and EDMA2

Feature	EDMA2	EDMA3
QDMA location	QDMA is part of the L2 cache controller that is a separate hardware module from EDMA channel and transfer controllers.	QDMA is an integral part of EDMA channel controller as shown in Figure 3.
PaRAM versus register for QDMA context saving	QDMA registers and pseudo registers for storing the contexts for its QDMA transfers.	Similar to DMA transfer, PaRAM entry table is used for storing the QDMA transfer context including synchronization type, source and destination addresses, etc.
QDMA event queue comparison	Event queue stores the actual transfer context from either PaRAM for DMA or from QDMA registers for QDMA transfer once the transfer request is admitted into event queue.	Two event queues are used for storing the indexes of DMA and QDMA events, rather than the actual context of PaRAM entries of the transfers. Only when the transfer request is submitted to EDMA transfer controller, the actual transfer context is loaded into the transfer controller.
	The difference is that on the EDMA2, once the transfer request is admitted into event queue, the QDMA registers are ready to be reconfigured by user for a new transfer. But on EDMA3, even after QDMA PaRAM set has been programmed, user still needs to wait until the transfer request is submitted into Transfer Controller to begin configuring the related PaRAM entry for a new transfer. Otherwise the content of the PaRAM entry of current QDMA transfer could be corrupted while the transfer request waiting in the event queue. User needs to be aware of this difference when configuring PaRAM for QDMA transfer on EDMA3.	
Number of QDMA channels	There is only one QDMA channel. Because the event queue on EDMA2 stores the actual transfer context and you can submit a new QDMA transfer once its current QDMA transfer request is admitted into the queue, EDMA2 can have multiple outstanding QDMA transfer requests waiting in the event queue. It's equivalent to having multiple QDMA channels. On EDMA2, you can allocate up to maximum 7 queue length of an event queue for QDMA. So equivalently, it has up to 7 QDMA channels.	8 QDMA channels. So at any given time, you can submit up to 8 outstanding QDMA requests.

2.6.2 QDMA Channel Mapping to PaRAM on EDMA3

On the EDMA3 similar to DMA transfer, the PaRAM entry table is used to specify QDMA transfer. As previously mentioned, each PaRAM entry has eight 32-bit words. The PAENTRY field in the QDMA channel mapping register (QCHMAP*n*) is used for mapping QDMA channel *n* to any one of the PaRAM entries as shown in Figure 7.

As mentioned in the previous section, QDMA on the EDMA2 does not use the PaRAM for storing transfer context. Rather, two sets of registers, QDMA and QDMA pseudo registers, are used for storing configuration information for QDMA transfer.

2.6.3 Triggering QDMA on EDMA3

On the EDMA3, you can specify a trigger word from any one of the eight 32-bit words of PaRAM for QDMA. Writing to the trigger word triggers the channel controller of QDMA to issue a transfer request. The trigger word (TRWORD) field in the QDMA channel mapping register (QCHMAP*n*) defines the trigger word for a particular QDMA channel *n* as shown in Figure 7.

This flexibility enables the CPU to selectively modify only the PaRAM entry that requires modification and thereby trigger the transfer. For example after a transfer, if only the count must change then QCHMAP*n* can be configured so count is the trigger word and a write to it automatically triggers the transfer.

On the EDMA2, writing to the QDMA pseudo registers leads to submitting a QDMA transfer request.

2.6.4 Linking on QDMA

On the EDMA3, linking is always enabled for both DMA and QDMA transfers. On the EDMA2, linking is not available for QDMA and the LINK address field in the channel options parameter (OPT) is reserved. See Section 2.10 for more detailed comparison between EDMA2 and EDMA3 linking.

Because linking is always enabled on EDMA3, when a QDMA transfer completes, its PaRAM is updated from its linked PaRAM entry and you are responsible to configure to terminate the linked operation for any QDMA operation that is enabled. You must set the LINK address field to FFFFh and the STATIC bit to 1 in OPT of the last PaRAM entry, to terminate a QDMA linked list.

2.6.5 Using IDMA for Efficient Configuration of QDMA PaRAM Sets on EDMA3

The internal DMA (IDMA) is a newly introduced DMA on DM644x devices and is part of the TMS320C64x+ DSP subsystem. IDMA performs fast block transfers between two linear ranges of memory locations local to the device's DSP subsystem. IDMA has a mask register that allows you to enable/disable transfers to specific words within the linear range. The local memory includes level 1 program (L1P), level 1 data (L1D), level 2 (L2) memories, and peripheral configuration registers. For IDMA details, see the *TMS320C64x+ DSP Megamodule Reference Guide* (SPRU871).

IDMA is used to efficiently configure the PaRAM for QDMA operations. For example if the last word of PaRAM set is configured as trigger word, then you can use IDMA to copy the entire content to the PaRAM set using IDMA. Upon the completion of IDMA transferring the QDMA, the QDMA event register (QER) is set and a transfer request in submitted.

IDMA is also used to efficiently configure multiple PaRAM entries for linking applications. For example in an application that requires linking multiple PaRAM entries for QDMA operation, you can allocate consecutive PaRAM entries for a particular QDMA. The last PaRAM entry (at the highest memory location) is used for the first QDMA transfer. You can use IDMA to write all the PaRAM entries allocated for the QDMA. Upon completion of transferring data to the last PaRAM entry, QDMA will be triggered and the linking to the rest of PaRAM will happen automatically.

2.6.6 STATIC Bit in OPT and QDMA on EDMA3

In EDMA3, the STATIC bit in the channel options parameter (OPT) is introduced.

When STATIC = 0, PaRAM entries are updated during the course of the transfer as well as the link is updated at the completion of a QDMA transfer. When STATIC = 1, none of the PaRAM entries are updated during a transfer and the linking is disabled when the transfer completes.

On EDMA3, linking is always enabled. You need to specify a valid address of a PaRAM entry table in the LINK field of each of its linked PaRAM entries to form a linked list. For all the linked PaRAM entries other than the last, you should configure the STATIC bit to 0 to allow PaRAM fields to be updated. To terminate a QDMA operation, it is required to set the STATIC bit and configure the LINK field to FFFFh in the last linked PaRAM entry of QDMA transfers.

If a particular transfer only requires one QDMA transfer, then the LINK field should be configured to FFFFh and the STATIC bit set to 1 for the PaRAM entry.

2.6.7 Software Use Model on EDMA3

On EDMA3 when a transfer request from either DMA or QDMA is admitted to an event queue, only the event number is queued. Only after the transfer request is submitted to the EDMA transfer controller, the associated PaRAM entry is submitted to the channel controller and you can reconfigure the PaRAM entry for a new QDMA transfer.

It is important to note that, when reprogramming the PaRAM entry of a QDMA, you must check and be sure that the last linked QDMA event has been submitted to the EDMA transfer controller. Otherwise, potentially the contents of the PaRAM entry of a QDMA transfer request could be corrupted while the request is still waiting in the event queue.

The issue is when you know that a linking operation of QDMA is complete and the associated PaRAM entry is used for new QDMA operations. Figure 19 shows an example of a linking operation and the how the values of the QDMA event register (QER) and the QDMA secondary event register (QSER) change throughout the entire linking operation. See the later part of this document for more details about the secondary event registers.

There are two methods for tracking the status of PaRAM resources for any DMA and QDMA operation as summarized:

- Wait for completion of interrupt. You can wait for the completion of a particular DMA and/or QDMA transfer to determine if the PaRAM is available for use for new data transfers.
- Poll for transfer request submission to the transfer controller. This method is particularly useful for QDMA linking operations. You can poll the status of the related bit in SER/ER for DMA or QSER/QER for QDMA and the STATIC bit in OPT to determine if the related transfer request is submitted to the transfer controller. If both SER/ER or QSER/QER bits are zeros and the STATIC bit equals 1, that means the last linked QDMA event has been submitted to the transfer controller and the related PaRAM is available to be configured for future use.

As shown in Figure 19, once a QDMA linking operation is started, the QDMA channel will not be available until after QSER:QER values become 0:0 and the value of the STATIC bit equals 1, which indicates that the last QDMA transfer is already submitted to the transfer controller.

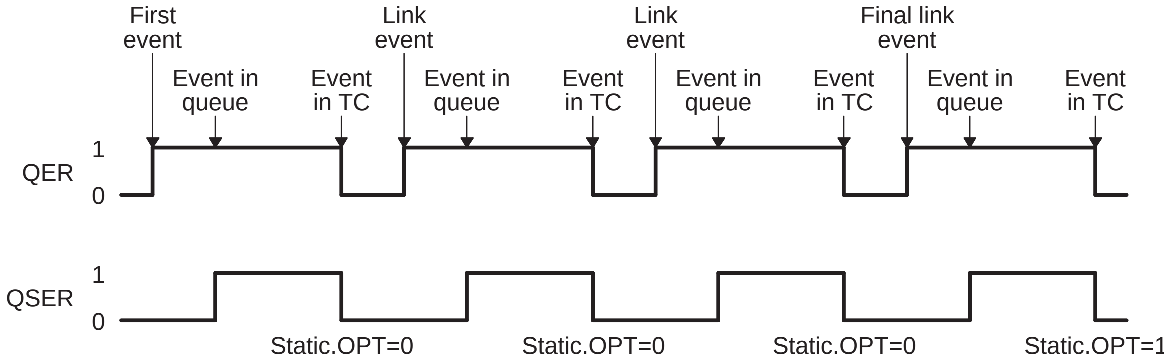


Figure 19. Linking Example with QER and QSER Values

2.7 *Interrupt on EDMA3*

On the high level, the interrupt operation on EDMA3 is similar to that on EDMA2. But there are some significant differences as summarized in this section.

2.7.1 Interrupt Types

The EDMA3 interrupts can be divided into 2 broad categories:

- Transfer completion interrupts
- Error interrupts

On DM644x devices, the transfer completion interrupts are listed in Table 5 and the error interrupts are listed in Table 6. For more information on the DSP interrupt controller (DSPINTC), see the *TMS320DM644x DMSoC DSP Subsystem Reference Guide* (SPRUE15). For more information on the ARM interrupt controller (AINTC), see the *TMS320DM644x DMSoC ARM Subsytem Reference Guide* (SPRUE14). In Table 5, shadow region 0, 1, and 2 are mentioned, see Section 2.7.3 for the concept of shadow region and shadow region interrupt.

Table 5. EDMA3 Transfer Completion Interrupts

				Interrupt Number	
Name	**Description**	**ARM**	**DSP**	**AINTC**	**DSPINTC**
EDMA3CC_INT0	EDMA3CC transfer completion interrupt shadow region 0	√	–	16	–

Table 5. EDMA3 Transfer Completion Interrupts (continued)

				Interrupt Number	
Name	**Description**	**ARM**	**DSP**	**AINTC**	**DSPINTC**
EDMA3CC_INT1	EDMA3CC transfer completion interrupt shadow region 1	–	√	–	36
EDMA3CC_INT2	EDMA3CC transfer completion interrupt shadow region 2	–	–	–	–

Table 6. EDMA3 Error Interrupts

				Interrupt Number	
Name	**Description**	**ARM**	**DSP**	**AINTC**	**DSPINTC**
EDMA3CC_ERRINT	EDMA3CC error interrupt	√	√	17	37
EDMA3TC0_ERRINT	TC0 error interrupt	√	√	18	38
EDMA3TC1_ERRINT	TC1 error interrupt	√	√	19	39

2.7.2 Comparison of Interrupt Registers Between EDMA2 and EDMA3

In both EDMA3 and EDMA2, there are equivalent parameters in the channel options parameter (OPT) and registers for interrupt operation. In addition on EDMA3, some new interrupt registers are introduced for interrupt. Table 7 summarizes the equivalent registers and parameters in OPT for interrupt operation on both EDMA3 and EDMA2. Table 8 summarizes the new registers that are introduced for interrupt operation on EDMA3. Table 9 summarizes the differences on interrupt parameters of OPT field and newly introduced interrupt registers on EDMA3.

Table 7. Equivalent Interrupt Parameters of OPT Field and Registers on EDMA2 and EDMA3

Interrupt Parameter	EDMA2	EDMA3
Transfer complete enable field in OPT	TCINT	TCINTEN
Transfer complete code field in OPT	TCCM:TCC	TCC
Intermediate transfer completion interrupt enable register	ATCINT	ITCINTEN
Intermediate transfer complete code register	ATCC	TCC
Interrupt pending registers	CIPRL, CIPRH	IPR, IPRH
Interrupt enable registers	CIERL, CIERH	IER, IERH

Table 8. New Pseudo-Interrupt Registers Introduced on EDMA3

Description	Name
Interrupt clear registers	ICR, ICRH
Interrupt enable set registers	IESR, IESRH
Interrupt enable clear registers	IECR, IECRH
Interrupt evaluate register	IEVAL

Table 9. Summary of Interrupt Parameters of OPT Field Differences Between EDMA3 and EDMA2

Feature	EDMA2	EDMA3
Transfer complete code	Two fields, 2-bit TCCM and 4-bit TCC field, to form the final 6-bit interrupt complete code.	Single 6-bit field for interrupt complete code.
Intermediate transfer complete code	Uses TCC bit for interrupt complete code and ATCC bit for intermediate interrupt complete code.	Uses the same TCC bit for both final and intermediate interrupt complete code.
Pseudo registers for interrupt enable register on EDMA3	–	Pseudo registers are introduced for interrupt enable registers. Instead of directly modifying the interrupt enable registers (IER and IERH), you must write to the interrupt enable set registers (IESR and IESRH) for enabling any interrupt and write to the interrupt enable clear registers (IECR and IECRH) for clearing any bits in interrupt enable registers.
Interrupt clear register (ICR and ICRH)	–	The interrupt clear registers (ICR and ICRH) are introduced for clearing any bits in the interrupt pending register (IPR).
Interrupt evaluate register (IEVAL)	–	The interrupt evaluate register (IEVAL) is introduced. A write of 1 to the EVAL bit in IEVAL forces the EDMA channel controller to generate an interrupt if there are still interrupts pending in IPR/IPRH. This feature needs to be used for writing robust DMA interrupt service routine.

2.7.3 Region Interrupt on EDMA3

On EDMA3, the concept of shadow region is introduced. Through each shadow region, you can only access a user-defined subset of DMA and QDMA resources. DMA and QDMA region control registers define or assign the ownership of DMA and/or QDMA channels for each shadow region. On DM644x devices, there are three shadow regions as shown in Table 10. For more details about shadow regions, see the *TMS320DM644x DMSoC Enhanced Direct Memory Access (EDMA3) Controller User's Guide* (SPRUE23).

Table 10. EDMA Shadow Regions

EDMA Master	Region	Region Access Registers
ARM	Shadow region 0	DRAE0 , DRAEH0 and QRAE0
DSP (C64x)	Shadow region 1	DRAE1, DRAEH1 and QRAE1
Image coprocessor	Shadow region 2	DRAE2, DRAEH2 and QRAE2

On DM644x devices, each shadow region has its own interrupt signal. The transfer completion of any DMA or QDMA channels belonging to a shadow region can potentially trigger the region interrupt signal for each of the shadow regions. Table 11 lists the interrupt signal for each region.

Table 11. EDMA Shadow Region Interrupt Signals

EDMA Master	Region	Region Interrupt Signal
ARM	Shadow region 0	EDMA3CC_INT0
DSP (C64x)	Shadow region 1	EDMA3CC_INT1
Image coprocessor	Shadow region 2	EDMA3CC_INT2

In addition to the global interrupt enable registers, the DMA and QDMA region access enable registers (DRAE*n*) serve as secondary interrupt enable registers. To enable the DMA transfer completion interrupt of a particular channel of a shadow region, both the corresponding bits in the global interrupt enable register and DMA region access enable register need to be set as shown:

- EDMA_CC_INT0 (to ARM):

```
(IPR.E0 & IER.E0 & DRAE0.E0) | (IPR.E1 & IER.E1 & DRAE0.E1) |…|(IPRH.E63 & IERH.E63 & DRAHE0.E63)
```

- EDMA_CC_INT1 (to DSP):

```
 (IPR.E0 & IER.E0 & DRAE1.E0) | (IPR.E1 & IER.E1 & DRAE1.E1) |…| (IPRH.E63 & IERH.E63 &
DRAHE1.E63)
```

- EDMA_CC_INT2 (to Image Coprocessor):

```
 (IPR.E0 & IER.E0 & DRAE2.E0) | (IPR.E1 & IER.E1 & DRAE2.E1) |…|(IPRH.E63 & IERH.E63 & DRAHE2.E63)
```

2.7.4 Interrupt Service Routine

On EDMA3, the channel controller does not generate a new interrupt signal for any new interrupts that are pending if previous pending interrupts have not been cleared by you. So software needs to manage to make sure all the pending interrupts are serviced by the interrupt service routines. In comparison, EDMA2 hardware is more forgiving. A write to clear a CIPR bit forces the EDMAINT signal to be repulsed if more interrupts are pending, thus minimizing the chance that interrupts are lost due to race conditions.

There are two options for constructing a robust interrupt service routine (ISR) for EDMA3:

- Poll all the bits during execution of the interrupt service routine (ISR), and clear all enabled bits in the interrupt pending register (IPR) by writing to the interrupt pending clear register (ICR/ICRH) before exiting any ISR as shown in the following pseudo code:
  1. Enter ISR
  2. Read IPR
  3. For the condition set in IPR,
     a. Perform operation as needed
     b. Clear bit for serviced INT
  4. Read IPR
     a. If IPR != 0, go to step 3
     b. If IPR == 0, exit ISR
- Service the interrupt service routine (ISR) and before exiting ISR, write to the EVAL bit in the interrupt evaluate register (IEVAL). This forces the EDMA channel controller to generate a new interrupt signal if there are still pending interrupts in the interrupt pending register (IPR/IPRH). The following lists the pseudo code for this ISR option:
  1. Enter ISR
  2. Read IPR
  3. For the condition set in IPR,
     a. Perform operation as needed
     b. Clear bit for serviced INT
  4. Read IPR
     a. If IPR == 0, exit ISR
     b. If IPR != 0, set EVAL bit in IEVAL to force to trigger a new interrupt

2.8 Event Queue

2.8.1 Number of Event Queues and Queue Length

On DM644x devices, EDMA3 has two event queues and each queue length is 16.

On C64x devices, EDMA2 has total of four event queues and each queue length is 16.

2.8.2 Available Queue Length for EDMA and QDMA

On EDMA2, the entire queue length is allocated for DMA, QDMA, and master peripherals and a maximum 7 queue length can be allocated to either DMA or QDMA.

On EDMA3, the entire queue length of 16 of all the available event queues is allocated for DMA and QDMA channels. So you do not need to manually configure for allocation of queue length for EDMA or QDMA on EDMA3.

2.8.3 Event Queue Timing

On EDMA3, when a transfer request from either DMA or QDMA is admitted to an event queue, only its event number is queued. Only after a transfer request is accepted into the transfer controller is the corresponding PaRAM free to be configured for future DMA transfer.

On EDMA2, the event queue stores the entire PaRAM entry table. So as soon as an event is admitted into the event queue, the PaRAM is ready for future use.

It is important that you be aware of the differences of the event queues between two DMAs. You must check and be sure the previous QDMA transfer request has been submitted to the EDMA transfer controller before configuring the related PaRAM for future use.

2.9 Priority Processing

2.9.1 Mapping Between DMA/QDMA Events and Event Queues

On EDMA3, the assignment or mapping of 64 DMA and 8 QDMA channels to two event queues of the channel controller is achieved by configuring the DMA queue number registers (DMAQNUM0-DMAQNUM1) and QDMA queue number register (QDMAQNUM).

On EDMA2, the four event queues are categorized into urgent, high priority, medium priority, and low priority queues. The mapping between DMA and QDMA channels and the four event queues is achieved by configuring the PRI parameter in the channel options parameter (OPT) of the corresponding PaRAM entry.

2.9.2 DMA/QDMA Queue Priorities versus Other Master Transfers

On EDMA3, only user-programmed DMA and QDMA transfers are handled by the EDMA transfer controller. The transfer requests eventually are submitted to two transfer controllers that are the masters of a switched central resource (SCR). Other transfers between different masters such as EMAC, DSP, and ARM to different slaves such as DDR and asynchronous EMIF directly go through the SCR. The priority arbitration between DMA/QDMA transfers and other master-to-slave transfers are handled by the SCR. The priorities of each of the masters can be programmed by you through the following registers:

- Event queue priority registers (QUEPRI0, QUEPRI1). These two registers are programmed by you to set the priorities of the two event queues of EDMA that are connected to the SCR through the EDMA transfer controller.
- Master priority registers (MSTRPRI0, MSTRPRI1). These two registers are programmed by you to set the priorities of masters including ARM, VLYNQ, ATA/CF, USB, and EMAC.

In contrary, on EDMA2 both DMA/QDMA transfers and other master-to-slave transfers go through the same event queues and transfer controller. All the transfers compete for the same bandwidth of the transfer controller. The priorities are programmed by configuring the PRI field of PaRAM to assign masters to different priority queues.

2.10 Linking on EDMA3

2.10.1 Linking for DMA

On EDMA3, linking for DMA and QDMA transfer is always enabled. You need to provide a valid link address in the LINK field of the corresponding PaRAM entry. In addition for linking on QDMA, you must clear the STATIC bit in the channel options parameter (OPT) to 0 for all the linked PaRAM entries except the last one and set to 1 for the last linked PaRAM entry.

On EDMA2, linking for DMA transfer is optional and can be enabled by setting LINK = 1 in OPT and provide a valid link address in the LINK field of the PaRAM entry. Linking is not available for QDMA on EDMA2. The link field of QDMA registers is reserved.

2.10.2 Terminating Linking

On both EDMA3 and EDMA2 linking is terminated by linking to a NULL parameter entry.

On EDMA3, you do not need to define a NULL entry explicitly and only need to set the LINK field of the last valid PaRAM entry to FFFFh. The channel controller automatically updates all the fields of the mapped PaRAM entry to zeros, except the LINK field that has the value of FFFFh. For QDMA, you must also set the STATIC bit in the channel options parameter (OPT) of the last valid PaRAM entry to 1.

On EDMA2, a NULL entry is defined as a PaRAM entry with all its fields being cleared to 0. A NULL entry must be defined and created by you. To terminate a linking, you need to link its last valid PaRAM entry to this NULL entry.

2.11 Chaining on EDMA3

Both EDMA3 and EDMA2 have similar chaining functionalities. If chaining is enabled for a DMA channel, the completion of the DMA transfer sets the chain event register of a DMA channel as specified by the TCC value in the channel options parameter (OPT) and thus triggers another transfer request. Although the chaining functionality is the same for both EDMA3 and EDMA2, there are some differences as summarized in Table 12.

On EDMA3, the transfer and intermediate transfer complete chaining enabling (TCCHEN and ITCCHEN) bits are different from those for transfer and intermediate transfer complete interrupt enabling (TCINTEN and ITCINTEN) bits. The TCCHEN and ITCCHEN bits in OPT are used for enabling transfer and intermediate transfer complete chaining. The TCINTEN and ITCINTEN bits in OPT are used for enabling transfer and intermediate transfer complete interrupts.

On EDMA2, the transfer and intermediate transfer complete chaining enabling (TCINT and ATCINT) bits are the same as those for transfer and intermediate transfer complete interrupt enabling bits. In addition, there are separate channel interrupt enable register (CIER) for enabling interrupt and chain enable register (CCER) for enabling chaining on EDMA2.

On EDMA3, having separate bits for enabling chaining and interrupt offers more flexibility for software development. The same TCC code can be used by two different DMA channels for different purposes. On one DMA channel, chaining can be enabled while the transfer completion interrupt is disabled. On the second DMA channel, transfer chaining can be disabled while its completion interrupt is enabled. This feature provides more flexibility and freedom for software development.

The same functionality of using the same TCC value for two different channels can not be achieved on EDMA2 because the same bits are used for both enabling chaining and interrupt.

Table 12. Comparison of Parameters for Chaining for EDMA3 and EDMA2

Feature	Description	EDMA2	EDMA3
Chaining/Interrupt	Chain enabling bit in OPT	TCINT	TCCHEN
	Interrupt enabling bit in OPT	TCINT	TCINTEN
	Transfer complete code	TCCM:TCC	TCC
Intermediate Chaining/Interrupt	Intermediate chain enabling bit in OPT	ATCINT	ITCCHEN
	Intermediate interrupt enabling bit in OPT	ATCINT	ITCINTEN
	Intermediate transfer complete code	ATCC	TCC

2.12 Set and Clear Operation on EDMA3

On EDMA3, pseudo registers are introduced for reliable operation in multitasking and multiprocessor environments. Instead of directly modifying interrupt and event registers, these registers are actually set or cleared through their pseudo registers. This new feature eliminates race conditions when multiple processors try to access the same DMA register at the same time. The following summarizes the pseudo registers that are introduced on EDMA3.

- EESR/EESRH and EECR/EECRH for the event enable register (EER/EERH). The event enable register (EER*n*/EERH*n*) is set through the event enable set register (EESR/EESRH) and is cleared through the event enable clear register (EECR/EECRH).
- QEESR/QEESRH and QEECR/QEECRH for the QDMA event enable register (QEER). The QDMA event enable register (QEER) is set through the QDMA set register (QEESR/QEESRH) and is cleared through the QDMA clear register (QEECR/QEECRH).
- IESR/IESRH and IECR/IECRH for the interrupt enable register (IER/IERH). The interrupt enable register (IER/IERH) is enabled through the interrupt set enable register (IESR/IESRH) and is cleared through the interrupt enable clear register (IECR/IECRH).

3 New Features on EDMA3

3.1 DMA Channel Region Access

The concept of DMA channel region access is introduced on EDMA3. Three types of regions are defined for the channel controller on EDMA3:

1. Global regions
2. Global DMA channel region
3. DMA channel shadow regions

The global region and global DMA channel region consist of all of the memory-mapped registers for all the DMA channels, QDMA channels, and interrupt.

The DMA channel shadow regions provide restricted view or access to the DMA and/or QDMA registers in the global DMA channel region. There are eight shadow regions on EDMA3. Each region has a region access enable register (DRAE*n*/DRAEH*n*) and QDMA region access enable register (QRAE*n*). The region access enable registers define the ownership or the right of access of the shadow regions to the DMA and QDMA channel registers in the global DMA channel region.

A value of 1 in a bit position on the DMA/QDMA region access enable register implies the ownership of that channel and that write accesses to the corresponding bit position are allowed to take effect and reads should return valid data. A value of 0 means that writes to that bit position are discarded without modifying the original register value and reads return a value of 0.

On DM644x devices, three shadow regions for ARM, DSP, and image coprocessor are defined. Table 10 summarizes the region access enable registers for each of the shadow region.

3.2 Debugging Capabilities

3.2.1 Debug Visibility

3.2.1.1 Event Queue Visibility

On EDMA3, the history of 16 events in every event queue can be monitored. The EDMA channel controller records the events in each event queue in the event queue entry registers (Q*k*E*n*), where *k* = 0-1 and *n* = 0-15, and *k* identifies the event queue and *n* indicates the order of the event entry.

The event queue entry registers capture the type of event, including the event triggered via ER, manual trigger via ESR, chain triggered via CER, and autotriggered via QER event, and the event number; or which of the DMA or QDMA channels.

3.2.1.2 Event Queue Watermarking

The event queue watermarking feature on EDMA3 monitors the maximum number of event entries in each of the event queues. The watermarking information is captured in the WM field of the event queue status register (QSTAT*n*).

3.2.1.3 Event Queue Threshold

The EDMA3 channel controller keeps track of whether the number of event entries in an event queue exceeds the user-programmed threshold. The queue threshold for each event queue is programmed in the queue threshold *x* registers (QWMTHRA and QWMTHRB) by you. A queue threshold error happens whenever the number of event entries in any of the event queue exceeds the user-programmed threshold and the error is captured in the THRXCD field of the event queue status register (QSTAT*n*) and the QTHRXCD*n* field in the EDMA3CC error register (CCERR).

3.2.2 Error Detection

3.2.2.1 Event Missed Registers

On EDMA3, the event miss register (EMR/EMRH) and the QDMA event miss register (QEMR) are introduced.

For EMR/EMRH, if any DMA channel event happens before the channel’s previous event has been cleared or submitted to the transfer controller, the event is considered a missed event and is captured in the corresponding bit of EMR or EMRH. You can clear the bits in EMR/EMRH by writing a 1 to the corresponding bit in the event miss clear register (EMCR/EMCRH).

For QEMR, if any QDMA channel event happens before the channel’s previous event has been cleared or submitted to the transfer controller, the event is considered a missed event and is captured in the corresponding bit of QEMR. You can clear the bits in EMR/EMRH by writing a 1 to the corresponding bit in the QDMA event miss clear register (QEMCR).

3.2.2.2 Channel Controller Error Registers

The EDMA3CC error register (CCERR) captures the queue threshold error and transfer completion code error.

3.2.2.3 Error from Null Parameter Entry

On EDMA3, an error occurs if an event is triggered for a NULL PaRAM entry. When this error happens, the EDMA channel controller sets both the corresponding bits in the secondary event register (SER) and event miss register (EMR).

When a 1 is set in SER, the channel controller disables the corresponding DMA or EDMA channel for future use. You need to clear both bits in SER and EMR in order to bring back the disabled DMA or QDMA channel.

3.2.2.4 Error Interrupts

Whenever there is a DMA missed event error, QDMA missed event error, and queue threshold error, the EDMA channel controller generates an interrupt signal via EDMA3CC_ERRINT.

Whenever there is a memory protection violation, the EDMA channel controller issues an interrupt signal via EDMA3CC_MPINT.

IMPORTANT NOTICE FOR TI DESIGN INFORMATION AND RESOURCES

Texas Instruments Incorporated (‘TI”) technical, application or other design advice, services or information, including, but not limited to, reference designs and materials relating to evaluation modules, (collectively, “TI Resources”) are intended to assist designers who are developing applications that incorporate TI products; by downloading, accessing or using any particular TI Resource in any way, you (individually or, if you are acting on behalf of a company, your company) agree to use it solely for this purpose and subject to the terms of this Notice.

TI’s provision of TI Resources does not expand or otherwise alter TI’s applicable published warranties or warranty disclaimers for TI products, and no additional obligations or liabilities arise from TI providing such TI Resources. TI reserves the right to make corrections, enhancements, improvements and other changes to its TI Resources.

You understand and agree that you remain responsible for using your independent analysis, evaluation and judgment in designing your applications and that you have full and exclusive responsibility to assure the safety of your applications and compliance of your applications (and of all TI products used in or for your applications) with all applicable regulations, laws and other applicable requirements. You represent that, with respect to your applications, you have all the necessary expertise to create and implement safeguards that (1) anticipate dangerous consequences of failures, (2) monitor failures and their consequences, and (3) lessen the likelihood of failures that might cause harm and take appropriate actions. You agree that prior to using or distributing any applications that include TI products, you will thoroughly test such applications and the functionality of such TI products as used in such applications. TI has not conducted any testing other than that specifically described in the published documentation for a particular TI Resource.

You are authorized to use, copy and modify any individual TI Resource only in connection with the development of applications that include the TI product(s) identified in such TI Resource. NO OTHER LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE TO ANY OTHER TI INTELLECTUAL PROPERTY RIGHT, AND NO LICENSE TO ANY TECHNOLOGY OR INTELLECTUAL PROPERTY RIGHT OF TI OR ANY THIRD PARTY IS GRANTED HEREIN, including but not limited to any patent right, copyright, mask work right, or other intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information regarding or referencing third-party products or services does not constitute a license to use such products or services, or a warranty or endorsement thereof. Use of TI Resources may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

TI RESOURCES ARE PROVIDED “AS IS” AND WITH ALL FAULTS. TI DISCLAIMS ALL OTHER WARRANTIES OR REPRESENTATIONS, EXPRESS OR IMPLIED, REGARDING TI RESOURCES OR USE THEREOF, INCLUDING BUT NOT LIMITED TO ACCURACY OR COMPLETENESS, TITLE, ANY EPIDEMIC FAILURE WARRANTY AND ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NON-INFRINGEMENT OF ANY THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

TI SHALL NOT BE LIABLE FOR AND SHALL NOT DEFEND OR INDEMNIFY YOU AGAINST ANY CLAIM, INCLUDING BUT NOT LIMITED TO ANY INFRINGEMENT CLAIM THAT RELATES TO OR IS BASED ON ANY COMBINATION OF PRODUCTS EVEN IF DESCRIBED IN TI RESOURCES OR OTHERWISE. IN NO EVENT SHALL TI BE LIABLE FOR ANY ACTUAL, DIRECT, SPECIAL, COLLATERAL, INDIRECT, PUNITIVE, INCIDENTAL, CONSEQUENTIAL OR EXEMPLARY DAMAGES IN CONNECTION WITH OR ARISING OUT OF TI RESOURCES OR USE THEREOF, AND REGARDLESS OF WHETHER TI HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

You agree to fully indemnify TI and its representatives against any damages, costs, losses, and/or liabilities arising out of your non-compliance with the terms and provisions of this Notice.

This Notice applies to TI Resources. Additional terms apply to the use and purchase of certain types of materials, TI products and services. These include; without limitation, TI’s standard terms for semiconductor products http://www.ti.com/sc/docs/stdterms.htm), evaluation modules, and samples (http://www.ti.com/sc/docs/sampterms.htm).

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265