

TMS320C6457 DSP Enhanced DMA (EDMA3) Controller

User's Guide



Literature Number: SPRUGK6
March 2009

| | |
|---|-----------|
| Preface | 13 |
| 1 Introduction | 15 |
| 1.1 Overview..... | 16 |
| 1.2 Features | 16 |
| 1.3 Terminology Used in This Document | 17 |
| 2 EDMA3 Architecture | 21 |
| 2.1 Functional Overview | 22 |
| 2.1.1 EDMA3 Controller Block Diagram..... | 22 |
| 2.1.2 EDMA3 Channel Controller (EDMA3CC) | 22 |
| 2.1.3 EDMA3 Transfer Controller (EDMA3TC)..... | 24 |
| 2.2 Types of EDMA3 Transfers | 25 |
| 2.2.1 A-Synchronized Transfers..... | 25 |
| 2.2.2 AB-Synchronized Transfers..... | 26 |
| 2.3 Parameter RAM (PaRAM) | 27 |
| 2.3.1 PaRAM | 28 |
| 2.3.2 EDMA3 Channel PaRAM Set Entry Fields..... | 30 |
| 2.3.3 Null PaRAM Set | 33 |
| 2.3.4 Dummy PaRAM Set..... | 33 |
| 2.3.5 Dummy Versus Null Transfer Comparison..... | 34 |
| 2.3.6 Parameter Set Updates | 34 |
| 2.3.7 Linking Transfers | 35 |
| 2.3.8 Fixed-Mode Transfers/Alignment Issues | 36 |
| 2.3.9 Element Size | 36 |
| 2.4 Initiating a DMA Transfer | 38 |
| 2.4.1 DMA Channel | 38 |
| 2.4.2 QDMA Channels | 41 |
| 2.4.3 Comparison Between DMA and QDMA Channels..... | 41 |
| 2.5 Completion of a DMA Transfer | 41 |
| 2.5.1 Normal Completion..... | 43 |
| 2.5.2 Early Completion | 43 |
| 2.5.3 Dummy or Null Completion | 43 |
| 2.6 Event, Channel, and PaRAM Mapping | 43 |
| 2.6.1 DMA Channel to PaRAM Mapping..... | 43 |
| 2.6.2 QDMA Channel to PaRAM Mapping..... | 44 |
| 2.7 EDMA3 Channel Controller Regions | 45 |
| 2.7.1 Region Overview | 45 |
| 2.7.2 Channel Controller Regions..... | 47 |
| 2.7.3 Region Interrupts | 47 |
| 2.8 Chaining EDMA3 Channels | 48 |
| 2.9 EDMA3 Interrupts | 49 |
| 2.9.1 Transfer Completion Interrupts | 49 |

| | | |
|----------|---|-----------|
| 2.9.2 | EDMA3 Interrupt Servicing | 52 |
| 2.9.3 | Interrupt Evaluation Operations | 53 |
| 2.9.4 | Error Interrupts..... | 54 |
| 2.10 | Memory Protection | 56 |
| 2.10.1 | Active Memory Protection | 56 |
| 2.10.2 | Proxied Memory Protection | 58 |
| 2.11 | Event Queue(s)..... | 59 |
| 2.11.1 | DMA/QDMA Channel to Event Queue Mapping | 60 |
| 2.11.2 | Queue RAM Debug Visibility | 60 |
| 2.11.3 | Queue Resource Tracking | 60 |
| 2.11.4 | Performance Considerations | 60 |
| 2.12 | EDMA3 Transfer Controller (EDMA3TC)..... | 61 |
| 2.12.1 | Architecture Details | 61 |
| 2.12.2 | Memory Protection | 63 |
| 2.12.3 | Error Generation..... | 63 |
| 2.12.4 | Debug Features | 63 |
| 2.12.5 | EDMA3TC Configuration | 64 |
| 2.13 | Event Dataflow..... | 64 |
| 2.14 | EDMA3 Prioritization | 64 |
| 2.14.1 | Channel Priority | 65 |
| 2.14.2 | Trigger Source Priority..... | 65 |
| 2.14.3 | Dequeue Priority..... | 65 |
| 2.14.4 | Transfer Controller Priority | 67 |
| 2.15 | Reset Considerations | 67 |
| 2.16 | Emulation Considerations | 67 |
| 3 | EDMA3 Transfer Examples | 69 |
| 3.1 | Block Move Example | 70 |
| 3.2 | Subframe Extraction Example | 72 |
| 3.3 | Data Sorting Example | 74 |
| 3.4 | Peripheral Servicing Example | 76 |
| 3.4.1 | Nonbursting Peripherals..... | 76 |
| 3.4.2 | Bursting Peripherals | 78 |
| 3.4.3 | Continuous Operation | 80 |
| 3.4.4 | Ping-Pong Buffering..... | 83 |
| 3.4.5 | Transfer Chaining Examples..... | 87 |
| 4 | Registers | 91 |
| 4.1 | Register Memory Maps..... | 91 |
| 4.2 | EDMA3 Channel Controller Control Registers | 91 |
| 4.2.1 | Global Registers..... | 94 |
| 4.2.2 | Error Registers | 102 |
| 4.2.3 | Region Access Enable Registers..... | 110 |
| 4.2.4 | Status/Debug Visibility Registers | 112 |
| 4.2.5 | Memory Protection Address Space | 118 |
| 4.2.6 | DMA Channel Registers | 121 |
| 4.2.7 | Interrupt Registers | 132 |
| 4.2.8 | QDMA Registers | 138 |
| 4.3 | EDMA3 Transfer Controller Control Registers | 144 |
| 4.3.1 | Peripheral Identification Register (PID) | 145 |

| | | |
|----------|--|------------|
| 4.3.2 | EDMA3TC Configuration Register (TCCFG)..... | 146 |
| 4.3.3 | EDMA3TC Channel Status Register (TCSTAT) | 147 |
| 4.3.4 | Error Registers | 149 |
| 4.3.5 | Read Rate Register (RDRATE)..... | 154 |
| 4.3.6 | EDMA3TC Channel Registers | 155 |
| A | Tips | 171 |
| A.1 | Debug Checklist..... | 171 |
| A.2 | Miscellaneous Programming/Debug Tips..... | 172 |
| B | Setting Up a Transfer | 173 |

List of Figures

| | | |
|------|--|-----|
| 2-1 | EDMA3 Controller Block Diagram..... | 22 |
| 2-2 | EDMA3 Channel Controller (EDMA3CC) Block Diagram | 23 |
| 2-3 | EDMA3 Transfer Controller (EDMA3TC) Block Diagram | 24 |
| 2-4 | Definition of ACNT, BCNT, and CCNT | 25 |
| 2-5 | A-Synchronized Transfers (ACNT = n, BCNT = 4, CCNT = 3)..... | 26 |
| 2-6 | AB-Synchronized Transfers (ACNT = n, BCNT = 4, CCNT = 3)..... | 27 |
| 2-7 | PaRAM Set..... | 28 |
| 2-8 | Channel Options Parameter (OPT) | 30 |
| 2-9 | Linked Transfer | 37 |
| 2-10 | DMA Channel and QDMA Channel to PaRAM Mapping..... | 44 |
| 2-11 | Shadow Region Registers | 46 |
| 2-12 | Interrupt Diagram..... | 51 |
| 2-13 | Error Interrupt Operation | 55 |
| 2-14 | User Write Access to EESR - MPPA Allows User Write Accesses..... | 57 |
| 2-15 | PaRAM Set Content for Proxied Memory Protection Example..... | 58 |
| 2-16 | Proxied Memory Protection Example | 59 |
| 2-17 | EDMA3 Prioritization | 66 |
| 3-1 | Block Move Example | 70 |
| 3-2 | Block Move Example PaRAM Configuration..... | 71 |
| 3-3 | Subframe Extraction Example | 72 |
| 3-4 | Subframe Extraction Example PaRAM Configuration | 73 |
| 3-5 | Data Sorting Example | 74 |
| 3-6 | Data Sorting Example PaRAM Configuration | 75 |
| 3-7 | Servicing Incoming McBSP Data Example | 76 |
| 3-8 | Servicing Incoming McBSP Data Example PaRAM Configuration | 77 |
| 3-9 | Servicing Peripheral Burst Example | 78 |
| 3-10 | Servicing Peripheral Burst Example PaRAM Configuration | 79 |
| 3-11 | Servicing Continuous McBSP Data Example..... | 80 |
| 3-12 | Servicing Continuous McBSP Data Example PaRAM Configuration..... | 81 |
| 3-13 | Servicing Continuous McBSP Data Example Reload PaRAM Configuration | 82 |
| 3-14 | Ping-Pong Buffering for McBSP Data Example | 84 |
| 3-15 | Ping-Pong Buffering for McBSP Example PaRAM Configuration..... | 85 |
| 3-16 | Ping-Pong Buffering for McBSP Example Pong PaRAM Configuration | 86 |
| 3-17 | Ping-Pong Buffering for McBSP Example Ping PaRAM Configuration | 86 |
| 3-18 | Intermediate Transfer Completion Chaining Example..... | 88 |
| 3-19 | Single Large Block Transfer Example | 88 |
| 3-20 | Smaller Packet Data Transfers Example..... | 89 |
| 4-1 | Peripheral ID Register (PID) | 94 |
| 4-2 | EDMA3CC Configuration Register (CCCFG)..... | 95 |
| 4-3 | DMA Channel Map <i>n</i> Registers (DCHMAP <i>n</i>)..... | 96 |
| 4-4 | QDMA Channel Map <i>n</i> Registers (QCHMAP <i>n</i>) | 97 |
| 4-5 | DMA Channel Queue <i>n</i> Number Registers (DMAQNUM <i>n</i>)..... | 98 |
| 4-6 | QDMA Channel Queue Number Register (QDMAQNUM)..... | 99 |
| 4-7 | Queue-to-TC Mapping Register (QUETCMAP)..... | 100 |
| 4-8 | Queue Priority Register (QUEPRI) | 101 |
| 4-9 | Event Missed Register (EMR)..... | 102 |
| 4-10 | Event Missed Register High (EMRH) | 102 |
| 4-11 | Event Missed Clear Register (EMCR) | 103 |
| 4-12 | Event Missed Clear Register High (EMCRH) | 103 |
| 4-13 | QDMA Event Missed Register (QEMR)..... | 104 |
| 4-14 | QDMA Event Missed Clear Register (QEMCR) | 105 |
| 4-15 | EDMA3CC Error Register (CCERR) | 106 |

| | | |
|------|---|-----|
| 4-16 | EDMA3CC Error Clear Register (CCERRCLR)..... | 108 |
| 4-17 | Error Evaluation Register (EEVAL)..... | 109 |
| 4-18 | DMA Region Access Enable Register for Region <i>m</i> (DRAEm)..... | 110 |
| 4-19 | DMA Region Access Enable High Register for Region <i>m</i> (DRAEHm) | 110 |
| 4-20 | QDMA Region Access Enable for Region <i>m</i> (QRAEm) | 111 |
| 4-21 | Event Queue Entry Registers (QxEy) | 112 |
| 4-22 | Queue Status Register <i>n</i> (QSTAT <i>n</i>) | 113 |
| 4-23 | Queue Watermark Threshold A Register (QWMTHRA) | 114 |
| 4-24 | Queue Watermark Threshold B Register (QWMTHRB) | 115 |
| 4-25 | EDMA3CC Status Register (CCSTAT) | 116 |
| 4-26 | Memory Protection Fault Address Register (MPFAR) | 118 |
| 4-27 | Memory Protection Fault Status Register (MPFSR) | 118 |
| 4-28 | Memory Protection Fault Command Register (MPFCR) | 119 |
| 4-29 | Memory Protection Page Attribute Register (MPPA[G/ <i>n</i>])..... | 120 |
| 4-30 | Event Register (ER) | 122 |
| 4-31 | Event Register High (ERH) | 122 |
| 4-32 | Event Clear Register (ECR) | 123 |
| 4-33 | Event Clear Register High (ECRH)..... | 123 |
| 4-34 | Event Set Register (ESR)..... | 124 |
| 4-35 | Event Set Register High (ESRH) | 125 |
| 4-36 | Chained Event Register (CER) | 126 |
| 4-37 | Chained Event Register High (CERH) | 126 |
| 4-38 | Event Enable Register (EER) | 127 |
| 4-39 | Event Enable Register High (EERH) | 127 |
| 4-40 | Event Enable Clear Register (EECR) | 128 |
| 4-41 | Event Enable Clear Register High (EECRH)..... | 128 |
| 4-42 | Event Enable Set Register (EESR) | 129 |
| 4-43 | Event Enable Set Register High (EESRH) | 129 |
| 4-44 | Secondary Event Register (SER)..... | 130 |
| 4-45 | Secondary Event Register High (SERH) | 130 |
| 4-46 | Secondary Event Clear Register (SECR) | 131 |
| 4-47 | Secondary Event Clear Register High (SECRH) | 131 |
| 4-48 | Interrupt Enable Register (IER) | 132 |
| 4-49 | Interrupt Enable Register High (IERH)..... | 132 |
| 4-50 | Interrupt Enable Clear Register (IECR) | 133 |
| 4-51 | Interrupt Enable Clear Register High (IECRH)..... | 133 |
| 4-52 | Interrupt Enable Set Register (IESR) | 134 |
| 4-53 | Interrupt Enable Set Register High (IESRH) | 134 |
| 4-54 | Interrupt Pending Register (IPR)..... | 135 |
| 4-55 | Interrupt Pending Register High (IPRH) | 135 |
| 4-56 | Interrupt Clear Register (ICR) | 136 |
| 4-57 | Interrupt Clear Register High (ICRH)..... | 136 |
| 4-58 | Interrupt Evaluate Register (IEVAL) | 137 |
| 4-59 | QDMA Event Register (QER) | 138 |
| 4-60 | QDMA Event Enable Register (QEER) | 139 |
| 4-61 | QDMA Event Enable Clear Register (QEECR) | 140 |
| 4-62 | QDMA Event Enable Set Register (QEESR) | 141 |
| 4-63 | QDMA Secondary Event Register (QSER)..... | 142 |
| 4-64 | QDMA Secondary Event Clear Register (QSECR) | 143 |
| 4-65 | Peripheral ID Register (PID)..... | 145 |
| 4-66 | EDMA3TC Configuration Register (TCCFG) | 146 |
| 4-67 | EDMA3TC Channel Status Register (TCSTAT) | 147 |
| 4-68 | Error Register (ERRSTAT) | 149 |

| | | |
|------|---|-----|
| 4-69 | Error Enable Register (ERREN) | 150 |
| 4-70 | Error Clear Register (ERRCLR) | 151 |
| 4-71 | Error Details Register (ERRDET)..... | 152 |
| 4-72 | Error Interrupt Command Register (ERRCMD)..... | 153 |
| 4-73 | Read Rate Register (RDRATE)..... | 154 |
| 4-74 | Source Active Options Register (SAOPT)..... | 156 |
| 4-75 | Source Active Source Address Register (SASRC)..... | 157 |
| 4-76 | Source Active Count Register (SACNT) | 157 |
| 4-77 | Source Active Destination Address Register (SADST) | 158 |
| 4-78 | Source Active Source B-Dimension Index Register (SABIDX) | 158 |
| 4-79 | Source Active Memory Protection Proxy Register (SAMPPTY)..... | 159 |
| 4-80 | Source Active Count Reload Register (SACNTRLD) | 160 |
| 4-81 | Source Active Source Address B-Reference Register (SASRCBREF)..... | 160 |
| 4-82 | Source Active Destination Address B-Reference Register (SADSTBREF) | 161 |
| 4-83 | Destination FIFO Options Register (DFOPT n)..... | 162 |
| 4-84 | Destination FIFO Source Address Register (DFSRC n)..... | 164 |
| 4-85 | Destination FIFO Count Register (DFCNT n)..... | 165 |
| 4-86 | Destination FIFO Destination Address Register (DFDST n) | 166 |
| 4-87 | Destination FIFO B-Index Register (DFBIDX n) | 166 |
| 4-88 | Destination FIFO Memory Protection Proxy Register (DFMPTY n)..... | 167 |
| 4-89 | Destination FIFO Count Reload Register (DFCNTRLD) | 168 |
| 4-90 | Destination FIFO Source Address B-Reference Register (DFSRCBREF)..... | 168 |
| 4-91 | Destination FIFO Destination Address B-Reference Register (DFDSTBREF) | 169 |

List of Tables

| | | |
|------|---|-----|
| 2-1 | EDMA3 Parameter RAM Contents..... | 27 |
| 2-2 | EDMA3 Channel Parameter Description..... | 28 |
| 2-3 | Channel Options Parameters (OPT) Field Descriptions..... | 30 |
| 2-4 | Dummy and Null Transfer Request..... | 34 |
| 2-5 | Parameter Updates in EDMA3CC (for Non-Null, Non-Dummy PaRAM Set)..... | 35 |
| 2-6 | EDMA3 Channel Synchronization Events..... | 38 |
| 2-7 | Expected Number of Transfers for Non-Null Transfer..... | 42 |
| 2-8 | Shadow Region Registers..... | 46 |
| 2-9 | Chain Event Triggers..... | 48 |
| 2-10 | EDMA3 Transfer Completion Interrupts..... | 49 |
| 2-11 | EDMA3 Error Interrupts..... | 49 |
| 2-12 | Number of Interrupts..... | 50 |
| 2-13 | Allowed Accesses..... | 56 |
| 2-14 | MPPA Registers to Region Assignment..... | 56 |
| 2-15 | Read/Write Command Optimization Rules..... | 61 |
| 2-16 | EDMA3 Transfer Controller Configurations..... | 64 |
| 4-1 | EDMACC Registers..... | 91 |
| 4-2 | Peripheral ID Register (PID) Field Descriptions..... | 94 |
| 4-3 | EDMA3CC Configuration Register (CCCFG) Field Descriptions..... | 95 |
| 4-4 | DMA Channel Map <i>n</i> Registers (DCHMAP <i>n</i>) Field Descriptions..... | 96 |
| 4-5 | QDMA Channel Map <i>n</i> Registers (QCHMAP <i>n</i>) Field Descriptions..... | 97 |
| 4-6 | DMA Channel Queue <i>n</i> Number Registers (DMAQNUM <i>n</i>) Field Descriptions..... | 98 |
| 4-7 | Bits in DMAQNUM <i>n</i> | 98 |
| 4-8 | QDMA Channel Queue Number Register (QDMAQNUM) Field Descriptions..... | 99 |
| 4-9 | Queue-to-TC Mapping Register (QUETCMAP) Field Descriptions..... | 100 |
| 4-10 | Queue Priority Register (QUEPRI) Field Descriptions..... | 101 |
| 4-11 | Event Missed Register (EMR) Field Descriptions..... | 102 |
| 4-12 | Event Missed Register High (EMRH) Field Descriptions..... | 103 |
| 4-13 | Event Missed Clear Register (EMCR) Field Descriptions..... | 103 |
| 4-14 | Event Missed Clear Register High (EMCRH) Field Descriptions..... | 103 |
| 4-15 | QDMA Event Missed Register (QEMR) Field Descriptions..... | 104 |
| 4-16 | QDMA Event Missed Clear Register (QEMCR) Field Descriptions..... | 105 |
| 4-17 | EDMA3CC Error Register (CCERR) Field Descriptions..... | 106 |
| 4-18 | EDMA3CC Error Clear Register (CCERRCLR) Field Descriptions..... | 108 |
| 4-19 | Error Evaluation Register (EEVAL) Field Descriptions..... | 109 |
| 4-20 | DMA Region Access Enable Registers for Region M (DRAEm/DRAEHm) Field Descriptions..... | 110 |
| 4-21 | QDMA Region Access Enable for Region M (QRAEm) Field Descriptions..... | 111 |
| 4-22 | Event Queue Entry Registers (QxEy) Field Descriptions..... | 112 |
| 4-23 | Queue Status Register <i>n</i> (QSTAT <i>n</i>) Field Descriptions..... | 113 |
| 4-24 | Queue Watermark Threshold A Register (QWMTHRA) Field Descriptions..... | 114 |
| 4-25 | Queue Watermark Threshold B Register (QWMTHRB) Field Descriptions..... | 115 |
| 4-26 | EDMA3CC Status Register (CCSTAT) Field Descriptions..... | 116 |
| 4-27 | Memory Protection Fault Address Register (MPFAR) Field Descriptions..... | 118 |
| 4-28 | Memory Protection Fault Status Register (MPFSR) Field Descriptions..... | 118 |
| 4-29 | Memory Protection Fault Command Register (MPFCR) Field Descriptions..... | 119 |
| 4-30 | Memory Protection Page Attribute Register (MPPA[G/ <i>n</i>]) Field Descriptions..... | 120 |
| 4-31 | Event Register (ER) Field Descriptions..... | 122 |
| 4-32 | Event Register High (ERH) Field Descriptions..... | 122 |
| 4-33 | Event Clear Register (ECR) Field Descriptions..... | 123 |

| | | |
|------|---|-----|
| 4-34 | Event Clear Register High (ECRH) Field Descriptions | 123 |
| 4-35 | Event Set Register (ESR) Field Descriptions | 124 |
| 4-36 | Event Set Register High (ESRH) Field Descriptions | 125 |
| 4-37 | Chained Event Register (CER) Field Descriptions | 126 |
| 4-38 | Chained Event Register High (CERH) Field Descriptions | 126 |
| 4-39 | Event Enable Register (EER) Field Descriptions | 127 |
| 4-40 | Event Enable Register High (EERH) Field Descriptions | 127 |
| 4-41 | Event Enable Clear Register (EECR) Field Descriptions | 128 |
| 4-42 | Event Enable Clear Register High (EECRH) Field Descriptions | 128 |
| 4-43 | Event Enable Set Register (EESR) Field Descriptions | 129 |
| 4-44 | Event Enable Set Register High (EESRH) Field Descriptions | 129 |
| 4-45 | Secondary Event Register (SER) Field Descriptions | 130 |
| 4-46 | Secondary Event Register High (SERH) Field Descriptions | 130 |
| 4-47 | Secondary Event Clear Register (SECR) Field Descriptions | 131 |
| 4-48 | Secondary Event Clear Register High (SECRH) Field Descriptions | 131 |
| 4-49 | Interrupt Enable Register (IER) Field Descriptions | 132 |
| 4-50 | Interrupt Enable Register High (IERH) Field Descriptions | 132 |
| 4-51 | Interrupt Enable Clear Register (IECR) Field Descriptions | 133 |
| 4-52 | Interrupt Enable Clear Register High (IECRH) Field Descriptions | 133 |
| 4-53 | Interrupt Enable Set Register (IESR) Field Descriptions | 134 |
| 4-54 | Interrupt Enable Set Register High (IESRH) Field Descriptions | 134 |
| 4-55 | Interrupt Pending Register (IPR) Field Descriptions | 135 |
| 4-56 | Interrupt Pending Register High (IPRH) Field Descriptions | 135 |
| 4-57 | Interrupt Clear Register (ICR) Field Descriptions | 136 |
| 4-58 | Interrupt Clear Register High (ICRH) Field Descriptions | 136 |
| 4-59 | Interrupt Evaluate Register (IEVAL) Field Descriptions | 137 |
| 4-60 | QDMA Event Register (QER) Field Descriptions | 138 |
| 4-61 | QDMA Event Enable Register (QEER) Field Descriptions | 139 |
| 4-62 | QDMA Event Enable Clear Register (QEECR) Field Descriptions | 140 |
| 4-63 | QDMA Event Enable Set Register (QEESR) Field Descriptions | 141 |
| 4-64 | QDMA Secondary Event Register (QSER) Field Descriptions | 142 |
| 4-65 | QDMA Secondary Event Clear Register (QSECR) Field Descriptions | 143 |
| 4-66 | EDMA3 Transfer Controller Registers | 144 |
| 4-67 | Peripheral ID Register (PID) Field Descriptions | 145 |
| 4-68 | EDMA3TC Configuration Register (TCCFG) Field Descriptions | 146 |
| 4-69 | EDMA3TC Channel Status Register (TCSTAT) Field Descriptions | 147 |
| 4-70 | Error Register (ERRSTAT) Field Descriptions | 149 |
| 4-71 | Error Enable Register (ERREN) Field Descriptions | 150 |
| 4-72 | Error Clear Register (ERRCLR) Field Descriptions | 151 |
| 4-73 | Error Details Register (ERRDET) Field Descriptions | 152 |
| 4-74 | Error Interrupt Command Register (ERRCMD) Field Descriptions | 153 |
| 4-75 | Read Rate Register (RDRATE) Field Descriptions | 154 |
| 4-76 | Source Active Options Register (SAOPT) Field Descriptions | 156 |
| 4-77 | Source Active Source Address Register (SASRC) Field Descriptions | 157 |
| 4-78 | Source Active Count Register (SACNT) Field Descriptions | 157 |
| 4-79 | Source Active Destination Address Register (SADST) Field Descriptions | 158 |
| 4-80 | Source Active Source B-Dimension Index Register (SABIDX) Field Descriptions | 158 |
| 4-81 | Source Active Memory Protection Proxy Register (SAMPPRXY) Field Descriptions | 159 |
| 4-82 | Source Active Count Reload Register (SACNTRLD) Field Descriptions | 160 |
| 4-83 | Source Active Source Address B-Reference Register (SASRCBREF) Field Descriptions | 160 |
| 4-84 | Source Active Destination Address B-Reference Register (SADSTBREF) Field Descriptions | 161 |

| | | |
|------|--|-----|
| 4-85 | Destination FIFO Options Register (DFOPT n) Field Descriptions | 162 |
| 4-86 | Destination FIFO Source Address Register (DFSRC n) Field Descriptions | 164 |
| 4-87 | Destination FIFO Count Register (DFCNT n) Field Descriptions | 165 |
| 4-88 | Destination FIFO Destination Address Register (DFDST n) Field Descriptions..... | 166 |
| 4-89 | Destination FIFO B-Index Register (DFBIDX n) Field Descriptions..... | 166 |
| 4-90 | Destination FIFO Memory Protection Proxy Register (DFMPPRXY n) Field Descriptions | 167 |
| 4-91 | Destination FIFO Count Reload Register (DFCNTRLD) Field Descriptions | 168 |
| 4-92 | Destination FIFO Source Address B-Reference Register (DFSRCBREF) Field Descriptions..... | 168 |
| 4-93 | Destination FIFO Destination Address B-Reference Register (DFDSTBREF) Field Descriptions | 169 |

Read This First

About This Manual

This document describes the Enhanced DMA (EDMA3) Controller in the TMS320C6457 digital signal processor (DSP).

Notational Conventions

This document uses the following conventions.

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

Related Documentation From Texas Instruments

The following documents describe the C6000™ devices and related support tools. Copies of these documents are available on the Internet at www.ti.com. *Tip:* Enter the literature number in the search box provided at www.ti.com.

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

[SPRU189](#) — *TMS320C6000 DSP CPU and Instruction Set Reference Guide*. Describes the CPU architecture, pipeline, instruction set, and interrupts for the TMS320C6000 digital signal processors (DSPs).

[SPRU198](#) — *TMS320C6000 Programmer's Guide*. Describes ways to optimize C and assembly code for the TMS320C6000™ DSPs and includes application program examples.

[SPRU301](#) — *TMS320C6000 Code Composer Studio Tutorial*. Introduces the Code Composer Studio™ integrated development environment and software tools.

[SPRU321](#) — *Code Composer Studio Application Programming Interface Reference Guide*. Describes the Code Composer Studio™ application programming interface (API), which allows you to program custom plug-ins for Code Composer.

[SPRU871](#) — *TMS320C64x+ DSP Megamodule Reference Guide*. Describes the TMS320C64x+ digital signal processor (DSP) megamodule. Included is a discussion on the internal direct memory access (IDMA) controller, the interrupt controller, the power-down controller, memory protection, bandwidth management, and the memory and cache.

Trademarks

C6000, TMS320C6000, Code Composer Studio are trademarks of Texas Instruments.

All other trademarks are the property of their respective owners.

Introduction

This document describes the features and operations of the enhanced direct memory access (EDMA3) controller in the TMS320C6457 device.

The EDMA3 is a high-performance, multichannel, multithreaded DMA controller that allows you to program a wide variety of transfer geometries and transfer sequences.

[Chapter 1](#) provides a overview of features and terminology. [Chapter 2](#) provides the architecture details and common operations of the EDMA3 channel controller (EDMA3CC) and the EDMA3 transfer controller (EDMA3TC). [Chapter 3](#) contains examples and common usage scenarios. [Chapter 4](#) describes the memory-mapped registers associated with the EDMA3 controller.

| Topic | Page |
|--|-------------|
| 1.1 Overview | 16 |
| 1.2 Features | 16 |
| 1.3 Terminology Used in This Document | 17 |

1.1 Overview

The enhanced direct memory access (EDMA3) controller's primary purpose is to service user-programmed data transfers between two memory-mapped slave endpoints on the device. Typical usage includes, but is not limited to:

- Servicing software driven paging transfers (for example, from external memory such as SDRAM to internal device memory such as DSP L2 SRAM)
- Servicing software or peripheral event driven data transfer between each of the DSP core L2 SRAM.
- Servicing event driven peripherals, such as a serial port peripheral like McBSP.
- Performing sorting or subframe extraction of various data structures.
- Offloading data transfers from the main device CPU(s) or DSP(s).
- For specific peripherals that are accessible via the EDMA3 controller, see the *TMS320C6457 Fixed-Point Digital Signal Processor* data manual ([SPRS582](#)).

The EDMA3 controller has a different architecture from the previous EDMA2 controller on the TMS320C621x/C671x DSPs and TMS320C64x DSPs. [For more details on new/advanced features, see the *EDMA v2.0 to EDMA v3.0 (EDMA3) Migration Guide* ([SPRAAP4](#)).]

The EDMA3 controller consists of two principal blocks:

- EDMA3 channel controller (EDMA3CC)
- EDMA3 transfer controller(s) (EDMA3TC)

The EDMA3 channel controller serves as the user interface for the EDMA3 controller. The EDMA3CC includes parameter RAM (PaRAM), channel control registers, and interrupt control registers. The EDMA3CC serves to prioritize incoming software requests or events from peripherals, and submits transfer requests (TRs) to the transfer controller.

The EDMA3 transfer controllers are slaves to the EDMA3 channel controller responsible for data movement. The transfer controller issues read/write commands to the source and destination addresses programmed for a given transfer. The operation is transparent to you.

1.2 Features

The EDMA3 channel controller has following features:

- Fully orthogonal transfer description
 - 3 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 3-D transfer based on 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:
 - Transfer completion
 - Error conditions
- Debug visibility
 - Queue watermarking/threshold
 - Error and status recording to facilitate debug
- 64 DMA channels
 - Event synchronization
 - Manual synchronization (CPU(s) write to event set register)

- Chain synchronization (completion of one transfer triggers another transfer)
- Eight QDMA channels
 - QDMA channels are triggered automatically upon writing to a PaRAM set entry
 - Support for programmable QDMA channel to PaRAM mapping
- 256 PaRAM sets
 - Each PaRAM set can be used for a DMA channel, QDMA channel, or link set
- Six transfer controllers/event queues. The system-level priority of these queues is user programmable. (For the possible system priorities, see the device-specific data manual.)
- 16 event entries per event queue
- Memory protection support
 - Proxied memory protection for TR submission
 - Active memory protection for accesses to PaRAM and registers

The EDMA3 transfer controller has the following features:

- Six transfer controllers (TC)
- All TCs have 128 bit wide read-write ports.
- Three transfer controllers with 64 bit wide read-write ports.
- Up to four in-flight transfer requests (TRs)
- Programmable priority level
- Supports 2-dimensional transfers with independent indexes on source and destination (EDMA3CC manages the 3rd dimension)
- Support for increment or FIFO mode transfers
- Interrupt and error support
- Little-endian or big-endian operation
- MMR bit fields are fixed position in 32-bit MMR regardless of endianness

1.3 Terminology Used in This Document

The following is a brief explanation of some terms used in this document:

| Term | Meaning |
|------------------------------|---|
| A-synchronized transfer | A transfer type where 1 dimension is serviced per synchronization event. |
| AB-synchronized transfer | A transfer type where 2 dimensions are serviced per synchronization event. |
| Chaining | A trigger mechanism in which a transfer can be initiated at the completion of another transfer or subtransfer. |
| CPU(s) | The main processing engine or engines on a device. Typically a DSP or general-purpose processor. (To learn more about the CPU on your system, see the device-specific data manual.) |
| Device | TMS320C6457 DSP. |
| DMA channel | One of the 64 channels that can be triggered by external, manual, and chained events. All DMA channels exist in the EDMA3CC. |
| Dummy set or dummy PaRAM set | A PaRAM set for which at least one of the count fields is equal to 0 and at least one of the count fields is nonzero. A null PaRAM set has all the count fields cleared. |
| Dummy transfer | A dummy set results in the EDMA3CC performing a dummy transfer. This is not an error condition. A null set results in an error condition. |

| Term | Meaning |
|--|---|
| EDMA3 channel controller (EDMA3CC) | The user-programmable portion of the EDMA3. The EDMA3CC contains the parameter RAM (PaRAM), event processing logic, DMA/QDMA channels, event queues, etc. The EDMA3CC services events (external, manual, chained, QDMA) and is responsible for submitting transfer requests to the transfer controllers (EDMA3TC), which perform the actual transfer. |
| EDMA3 programmer | Any entity on the chip that has read/write access to the EDMA3 registers and can program an EDMA3 transfer. |
| EDMA3 transfer controller(s) (EDMA3TC) | Transfer controllers are the transfer engines for the EDMA3 controller. They perform the read/writes as dictated by the transfer requests submitted by the EDMA3CC. |
| Enhanced direct memory access (EDMA3) controller | Consists of the EDMA3 channel controller (EDMA3CC) and EDMA3 transfer controller(s) (EDMA3TC), referred to as EDMA3 in this document. |
| Link parameter set | A PaRAM set that is used for linking. |
| Linking | The mechanism of reloading a PaRAM set with new transfer characteristics on completion of the current transfer. |
| Memory-mapped slave | All on-chip memories, off-chip memories, and slave peripherals. These typically rely on the EDMA3 (or other master peripheral) to perform transfers to and from them. |
| Master peripherals | All peripherals that are capable of initiating read and write transfers to the system and may not solely rely on the EDMA3 for their data transfers. |
| Null set or null PaRAM set | A PaRAM set has all count fields cleared (except for the link field). A dummy PaRAM set has at least one of the count fields nonzero. |
| Null transfer | A trigger event for a null PaRAM set. Results in the EDMA3CC performing a null transfer. This is an error condition. A dummy transfer does not result in an error. |
| Parameter RAM (PaRAM) | Programmable RAM that stores PaRAM sets used by DMA channels, QDMA channels, and linking. |
| Parameter RAM (PaRAM) set | A 32-byte EDMA3 channel transfer definition. Each parameter set consists of 8 words (4-bytes each), which store the context for a DMA/QDMA/link transfer. A PaRAM set includes source address, destination address, counts, indexes, options, etc. |
| Parameter RAM (PaRAM) set entry | One of the eight 4-byte components of the parameter set. |
| QDMA channel | One of the 8 channels that can be triggered when writing to the trigger word (TRWORD) of a PaRAM set. All QDMA channels exist in the EDMA3CC. |
| Slave end points | All on-chip memories, off-chip memories, and slave peripherals. These may rely on the EDMA3 to perform transfers to and from them. |
| Transfer request (TR) | A command for data movement that is issued from the EDMA3CC to the EDMA3TC. A TR includes source and destination addresses, counts, indexes, options, etc. |
| Trigger event | Action that causes the EDMA3CC to service the channel and submit a transfer request to the EDMA3TC. Trigger events for DMA channels include manual triggered (CPU triggered), external event triggered, and chain triggered. Trigger events for QDMA channels include automatically triggered and link triggered. |

| Term | Meaning |
|---------------------------------|---|
| Trigger word | For QDMA channels, the trigger word specifies the PaRAM set entry that results in a QDMA trigger event when written. The trigger word is programmed via the QDMA channel map register (QCHMAP) and can point to any of the PaRAM set entries. |
| TR synchronization (sync) event | See Trigger event. |

EDMA3 Architecture

This chapter discusses the architecture of the EDMA3 controller.

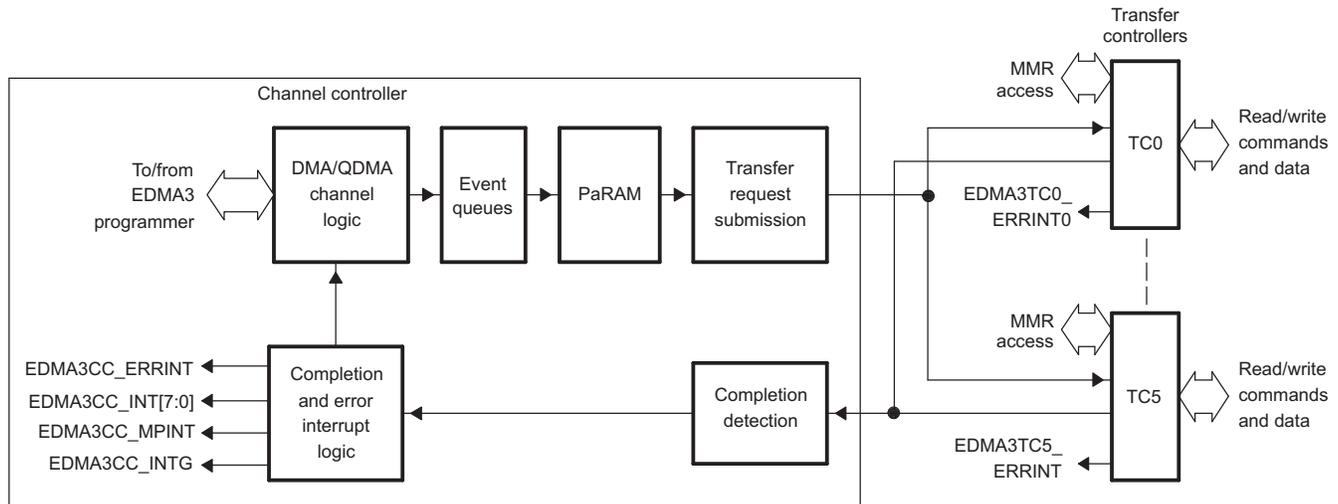
| Topic | Page |
|---|------|
| 2.1 Functional Overview..... | 22 |
| 2.2 Types of EDMA3 Transfers..... | 25 |
| 2.3 Parameter RAM (PaRAM) | 27 |
| 2.4 Initiating a DMA Transfer | 38 |
| 2.5 Completion of a DMA Transfer..... | 41 |
| 2.6 Event, Channel, and PaRAM Mapping..... | 43 |
| 2.7 EDMA3 Channel Controller Regions..... | 45 |
| 2.8 Chaining EDMA3 Channels | 48 |
| 2.9 EDMA3 Interrupts..... | 49 |
| 2.10 Memory Protection..... | 56 |
| 2.11 Event Queue(s) | 59 |
| 2.12 EDMA3 Transfer Controller (EDMA3TC)..... | 61 |
| 2.13 Event Dataflow..... | 64 |
| 2.14 EDMA3 Prioritization | 64 |
| 2.15 Reset Considerations | 67 |
| 2.16 Emulation Considerations..... | 67 |

2.1 Functional Overview

2.1.1 EDMA3 Controller Block Diagram

Figure 2-1 shows a block diagram for the EDMA3 controller.

Figure 2-1. EDMA3 Controller Block Diagram



2.1.2 EDMA3 Channel Controller (EDMA3CC)

Figure 2-2 shows a functional block diagram of the EDMA3 channel controller (EDMA3CC).

The main blocks of the EDMA3CC are:

- **Parameter RAM (PaRAM):** This 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. EDMA3CC processes sets based on a trigger event and submits a transfer request (TR) to the transfer controller.
- **EDMA3 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 EDMA3TC and/or slave peripherals. Completion of transfers can optionally be used to chain trigger new transfers or to assert interrupts.
- **Event queues:** These form the interface between the event detection logic and the transfer request submission logic.
- **Memory protection registers:** These registers define the accesses (privilege level and requestor(s)) allowed to access the DMA channel shadow region view(s) and regions of PaRAM.

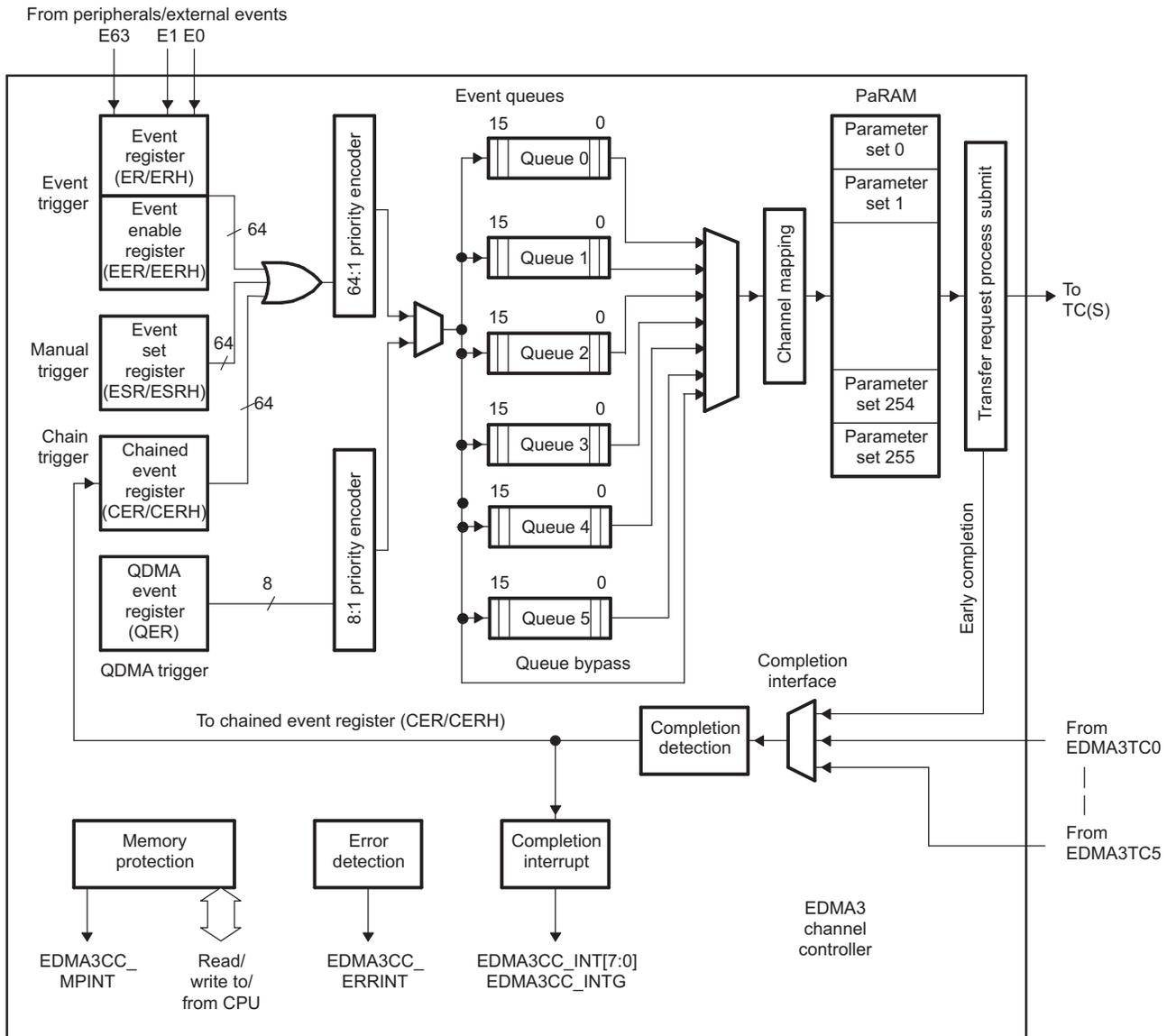
Other functions include:

- **Region registers:** Allows DMA resources (DMA channels and interrupts) to be assigned to unique regions, which are owned by different EDMA3 programmers (for example, DSP).
- **Debug registers:** Allows debug visibility by providing registers to read the queue status, controller status, and missed event status.

The EDMA3CC 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 QDMA channel is how the transfers are triggered by the system (see [Section 2.4](#)).

Figure 2-2. EDMA3 Channel Controller (EDMA3CC) Block Diagram



A trigger event is needed to initiate a transfer. For DMA channels, a trigger event may be due to an external event, manual write to the event set register, or chained event. QDMA channels are automatically triggered when a write is performed to the trigger word you programmed for the associated PaRAM set. All such trigger events are logged into appropriate registers upon recognition (see Section 4.2.1.3).

Once a trigger event is recognized, the event type/channel is queued in the appropriate EDMA3CC event queue. The assignment of each DMA/QDMA channel to an event queue is programmable. Each queue is 16 deep, so up to 16 events may be queued (on a single queue) in the EDMA3CC at a time. Additional pending events mapped to a full queue are queued when event queue space becomes available (see Section 2.11).

If events on different channels are detected simultaneously, the events are queued based on fixed priority arbitration scheme with the DMA channels being higher priority than the QDMA channels. Among the two groups of channels, the lowest-numbered channel is the highest priority.

Each event in the event queue is processed in the order it was queued. On reaching the head of the queue, the PaRAM associated with that channel is read to determine the transfer details. The TR submission logic evaluates the validity of the TR and is responsible for submitting a valid transfer request (TR) to the appropriate EDMA3TC (based on the event queue to EDMA3TC association, Q0 goes to TC0, Q1 goes to TC1, Q2 goes to TC2, Q3 goes to TC3, Q4 goes to TC4 and Q5 goes to TC5). For more details, see [Section 2.3](#).

The EDMA3TC receives the request and is responsible for of data movement as specified in the transfer request packet (TRP) and other necessary tasks like buffering, and ensuring transfers are carried out in an optimal fashion wherever possible. For more details on EDMA3TC, see [Section 2.1.3](#).

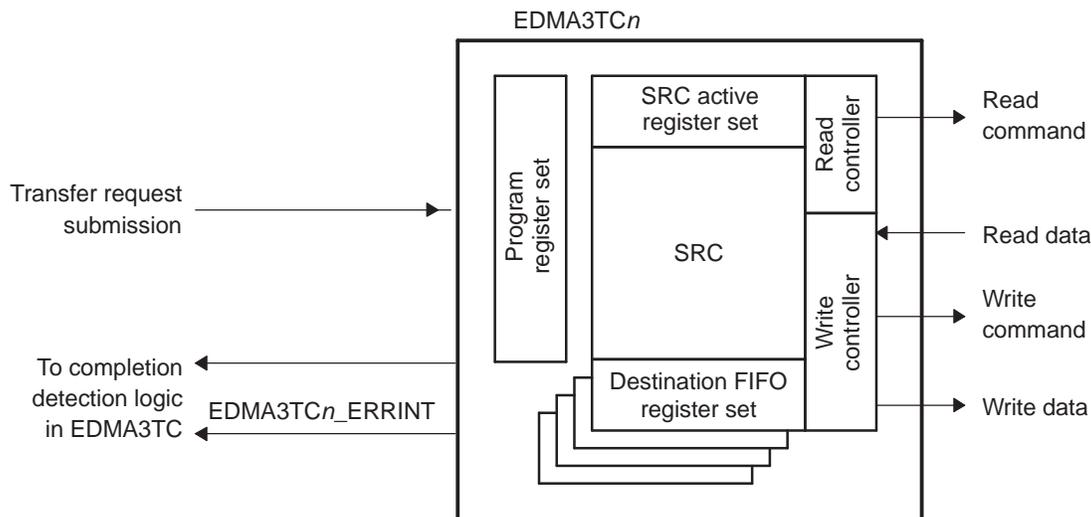
If you have decided to receive an interrupt or to chain to another channel on completion of the current transfer, the EDMA3TC signals completion to the EDMA3CC completion detection logic when the transfer is done. You can alternately choose to trigger completion when a TR leaves the EDMA3CC boundary rather than wait for all the data transfer to complete. Based on the setting of the EDMA3CC interrupt registers, the completion interrupt generation logic is responsible for generating EDMA3CC completion interrupts to the CPU. For more details, see [Section 2.5](#).

Additionally, the EDMA3CC also has an error detection logic, which causes an error interrupt generation on various error conditions (like missed events, exceeding event queue thresholds, etc.). For more details on error interrupts, see [Section 2.9.4](#).

2.1.3 EDMA3 Transfer Controller (EDMA3TC)

[Figure 2-3](#) shows a functional block diagram of the EDMA3 transfer controller (EDMA3TC).

Figure 2-3. EDMA3 Transfer Controller (EDMA3TC) Block Diagram



The main blocks of the EDMA3TC are:

- DMA program register set: The DMA program register set stores the transfer requests received from the EDMA3 channel controller (EDMA3CC).
- DMA source active register set: The DMA source active register set stores the context for the DMA transfer request currently in progress in the read controller.
- Read controller: The read controller issues read commands to the source address.
- Destination FIFO register set: The destination (DST) FIFO register set stores the context for the DMA transfer request(s) currently in progress in the write controller.
- Write controller: The write controller issues write commands/write data to the destination slave.
- Data FIFO: The data FIFO exists for holding temporary in-flight data.
- Completion interface: The completion interface sends completion codes to the EDMA3CC when a transfer completes, and generates interrupts and chained events (also, for details on transfer completion reporting, see [Section 2.1.2](#)).

When the EDMA3TC is idle and receives its first TR, the TR is received in the DMA program register set, where it transitions to the DMA source active set and the destination FIFO register set immediately. The second TR (if pending from EDMA3CC) is loaded into the DMA program set, ensuring it can start as soon as possible when the active transfer is completed. As soon as the current active set is exhausted, the TR is loaded from the DMA program register set into the DMA source active register set as well as to the appropriate entry in the destination FIFO register set.

The read controller issues read commands governed by the rules of command fragmentation and optimization. These are issued only when the data FIFO has space available for the 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. For details on command fragmentation and optimization, see [Section 2.12.1.1](#).

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.

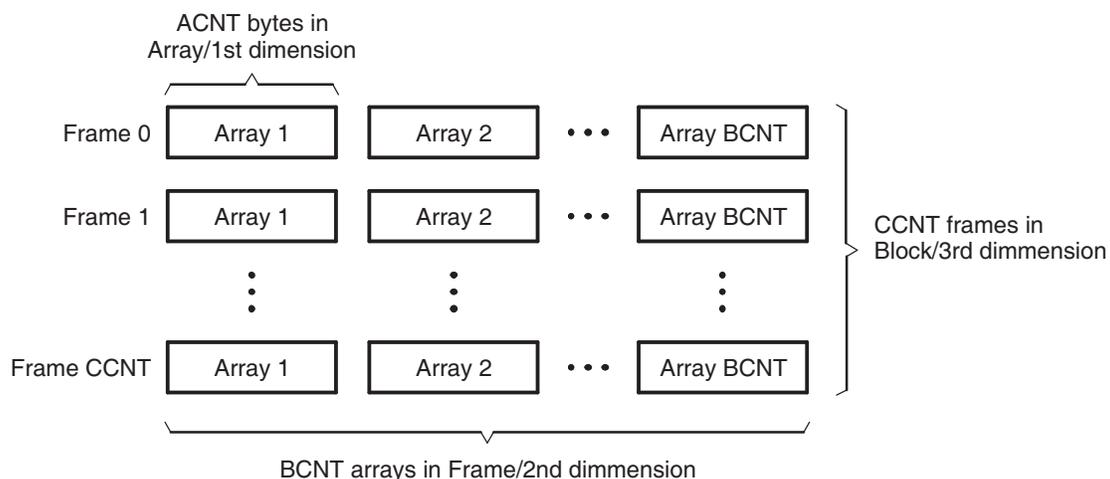
2.2 Types of EDMA3 Transfers

An EDMA3 transfer is always defined in terms of three dimensions. [Figure 2-4](#) shows the three dimensions used by EDMA3 transfers. These three dimensions are defined as:

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

Note that the reference point for the index depends on the synchronization type. The amount of data transferred upon receipt of a trigger/synch event is controlled by the synchronization types (SYNCDIM bit in OPT). Of the three dimensions, only two synchronization types are supported: A-synchronized transfers and AB-synchronized transfers.

Figure 2-4. Definition of ACNT, BCNT, and CCNT



2.2.1 A-Synchronized Transfers

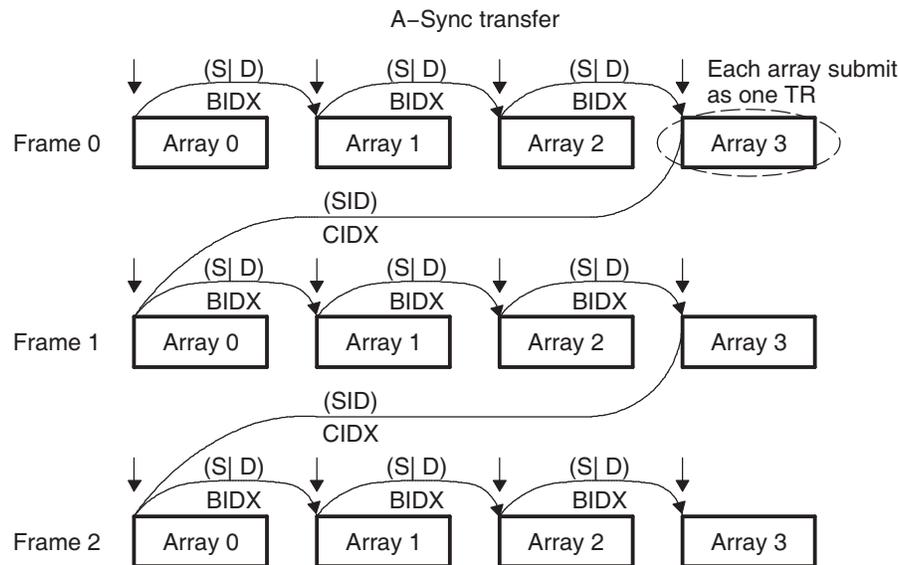
In an A-synchronized transfer, each EDMA3 sync event initiates the transfer of the 1st dimension of ACNT bytes, or one array of ACNT bytes. In other words, each event/TR packet conveys the transfer information for one array only. Thus, $BCNT \times CCNT$ events are needed to completely service a PaRAM set.

Arrays are always separated by SRCBIDX and DSTBIDX, as shown in [Figure 2-5](#), where the start address of Array N is equal to the start address of Array N - 1 plus source (S) or destination (D) BIDX.

Frames are always separated by SRCCIDX and DSTCIDX. For A-synchronized transfers, after the frame is exhausted, the address is updated by adding (S/D)CIDX to the beginning address of the last array in the frame. As in [Figure 2-5](#), SRCCIDX/DSTCIDX is the difference between the start of Frame 0 Array 3 to the start of Frame 1 Array 0.

[Figure 2-5](#) shows an A-synchronized transfer of 3 (CCNT) frames of 4 (BCNT) arrays of n (ACNT) bytes. In this example, a total of 12 sync events (BCNT × CCNT) exhaust a PaRAM set. For details on parameter set updates, see [Section 2.3.6](#).

Figure 2-5. A-Synchronized Transfers (ACNT = n, BCNT = 4, CCNT = 3)



2.2.2 AB-Synchronized Transfers

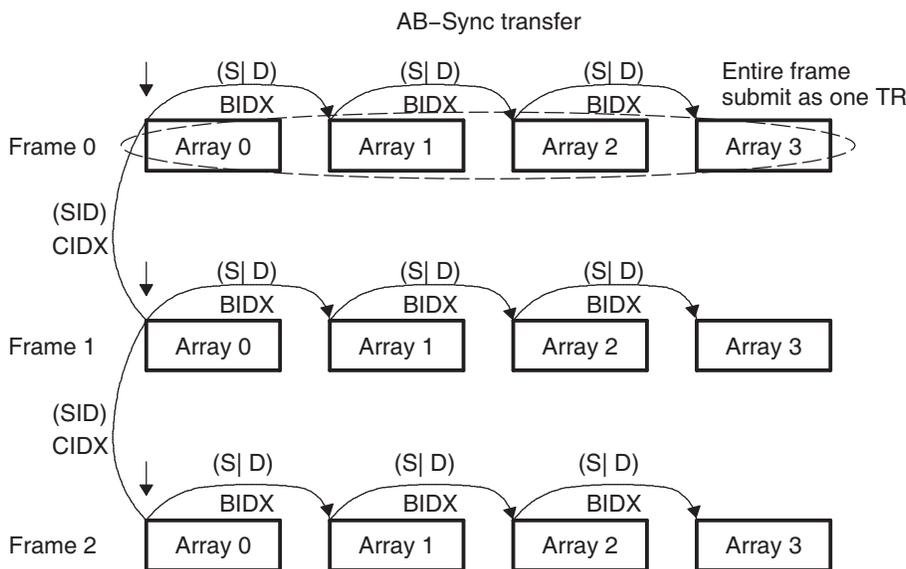
In a AB-synchronized transfer, each EDMA3 sync event initiates the transfer of 2 dimensions or one frame. In other words, each event/TR packet conveys information for one entire frame of BCNT arrays of ACNT bytes. Thus, CCNT events are needed to completely service a PaRAM set.

Arrays are always separated by SRCBIDX and DSTBIDX as shown in [Figure 2-6](#). Frames are always separated by SRCCIDX and DSTCIDX.

Note that for AB-synchronized transfers, after a TR for the frame is submitted, the address update is to add (S/D)CIDX 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 (S/D)CIDX to the beginning address of the last array in the frame. For details on parameter set updates, see [Section 2.3.6](#).

[Figure 2-6](#) shows an AB-synchronized transfer of three (CCNT) frames of four (BCNT) arrays of n (ACNT) bytes. In this example, a total of three sync events (CCNT) exhaust a PaRAM set, which is equivalent to a total of three transfers of four arrays each completes the transfer.

Figure 2-6. AB-Synchronized Transfers (ACNT = n, BCNT = 4, CCNT = 3)



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

2.3 Parameter RAM (PaRAM)

The EDMA3 controller is a RAM-based architecture. The transfer context (source/destination addresses, count, indexes, etc.) for DMA or QDMA channels is programmed in a parameter RAM table within EDMA3CC, referred to as PaRAM. The PaRAM table is segmented into multiple PaRAM sets. Each PaRAM set includes eight 4-byte PaRAM set entries (32-bytes total per PaRAM set), which includes typical DMA transfer parameters such as source address, destination address, transfer counts, indexes, options, etc.

The PaRAM structure supports flexible ping-pong, circular buffering, channel chaining, and automatic reloading (linking). The contents of the PaRAM include:

- 256 PaRAM sets
- Any PaRAM entries can be used for DMA, QDMA, or link sets
- By default, all channels map to PaRAM set 0. These should be remapped before use. For more information, see [Section 4.2.1.3](#) (DCHMAP registers) and [Section 4.2.1.4](#) (QCHMAP registers).

Table 2-1. EDMA3 Parameter RAM Contents

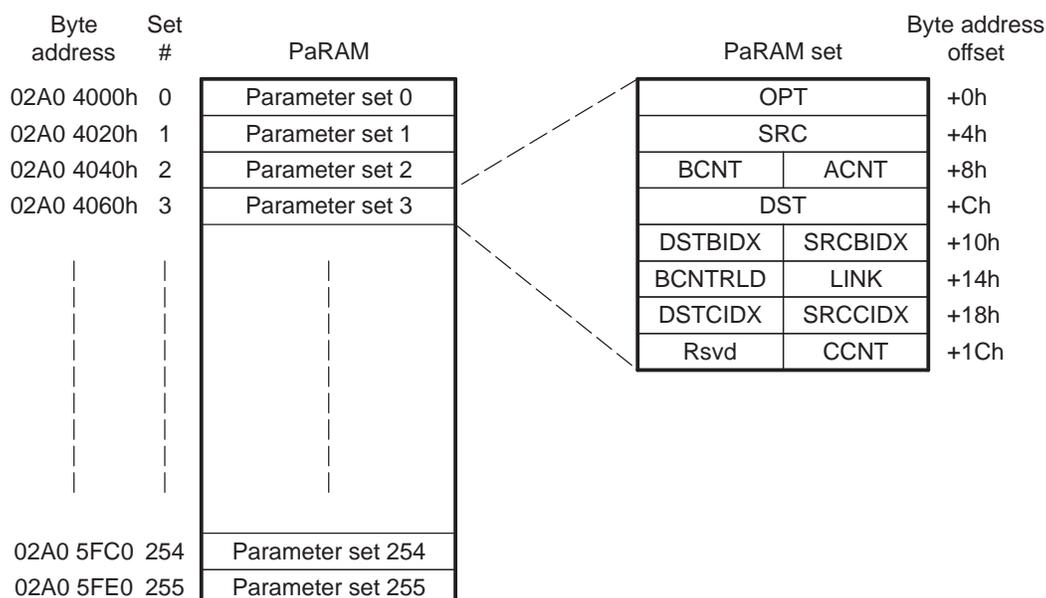
| PaRAM Set Number | Address | Parameters |
|------------------|--------------------------|-------------|
| 0 | 02A0 4000h to 02A0 401Fh | PaRAM set 0 |
| 1 | 02A0 4020h to 02A0 403Fh | PaRAM set 1 |
| 2 | 02A0 4040h to 02A0 405Fh | PaRAM set 2 |
| 3 | 02A0 4060h to 02A0 407Fh | PaRAM set 3 |
| 4 | 02A0 4080h to 02A0 409Fh | PaRAM set 4 |
| 5 | 02A0 40A0h to 02A0 40BFh | PaRAM set 5 |
| 6 | 02A0 40C0h to 02A0 40DFh | PaRAM set 6 |
| 7 | 02A0 40E0h to 02A0 40FFh | PaRAM set 7 |
| 8 | 02A0 4100h to 02A0 411Fh | PaRAM set 8 |
| 9 | 02A0 4120h to 02A0 413Fh | PaRAM set 9 |

Table 2-1. EDMA3 Parameter RAM Contents (continued)

| PaRAM Set Number | Address | Parameters |
|------------------|--------------------------|---------------|
| ... | ... | ... |
| 63 | 02A0 47E0h to 02A0 47FFh | PaRAM set 63 |
| 64 | 02A0 4800h to 02A0 481Fh | PaRAM set 64 |
| 65 | 02A0 4820h to 02A0 483Fh | PaRAM set 65 |
| ... | ... | ... |
| 254 | 02A0 5FC0h to 02A0 5FDFh | PaRAM set 254 |
| 255 | 02A0 5FE0h to 02A0 5FFFh | PaRAM set 255 |

2.3.1 PaRAM

Each parameter set of PaRAM is organized into eight 32-bit words or 32 bytes, as shown in [Figure 2-7](#) and described in [Table 2-2](#). Each PaRAM set consists of 16-bit and 32-bit parameters.

Figure 2-7. PaRAM Set

Table 2-2. EDMA3 Channel Parameter Description

| Offset Address (bytes) | Acronym | Parameter | Description |
|------------------------|---------|-----------------------------|---|
| 0h | OPT | Channel Options | Transfer configuration options |
| 4h | SRC | Channel Source Address | The byte address from which data is transferred |
| 8h ⁽¹⁾ | ACNT | Count for 1st Dimension | Unsigned value specifying the number of contiguous bytes within an array (first dimension of the transfer). Valid values range from 1 to 65 535. |
| | BCNT | Count for 2nd Dimension | Unsigned value specifying the number of arrays in a frame, where an array is ACNT bytes. Valid values range from 1 to 65 535. |
| Ch | DST | Channel Destination Address | The byte address to which data is transferred |
| 10h ⁽¹⁾ | SRCBIDX | Source BCNT Index | Signed value specifying the byte address offset between source arrays within a frame (2nd dimension). Valid values range from -32 768 and 32 767. |

⁽¹⁾ It is recommended to access the parameter set sets as 32-bit words whenever possible.

Table 2-2. EDMA3 Channel Parameter Description (continued)

| Offset Address (bytes) | Acronym | Parameter | Description |
|------------------------|---------|-------------------------|---|
| | DSTBIDX | Destination BCNT Index | The byte address offset of arrays within a frame (2nd dimension) for destination |
| 14h ⁽¹⁾ | LINK | Link Address | The PaRAM address containing the PaRAM set to be linked (copied from) when the current PaRAM set is exhausted. A value of FFFFh specifies a null link. |
| | BCNTRLD | BCNT Reload | The count value used to reload BCNT when BCNT decrements to 0 (TR is submitted for the last array in 2nd dimension). Only relevant in A-synchronized transfers. |
| 18h ⁽¹⁾ | SRCCIDX | Source CCNT Index | <p>Signed value specifying the byte address offset between frames within a block (3rd dimension). Valid values range from -32 768 and 32 767.</p> <p>A-synchronized transfers: Index is between the start address of the last source array in a frame to the start address of the first source array in the next frame.</p> <p>AB-synchronized transfers: Index is between the start address of the first source array in a frame to the start address of the first source array in the next frame.</p> |
| | DSTCIDX | Destination CCNT index | <p>Signed value specifying the byte address offset between frames within a block (3rd dimension). Valid values range from -32 768 and 32 767.</p> <p>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.</p> <p>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.</p> |
| 1Ch | CCNT | Count for 3rd Dimension | Unsigned value specifying the number of frames in a block, where a frame is BCNT arrays of ACNT bytes. Valid values range from 1 to 65 535. |
| | RSVD | Reserved | Reserved |

2.3.2 EDMA3 Channel PaRAM Set Entry Fields

2.3.2.1 Channel Options Parameter (OPT)

The channel options parameter (OPT) is shown in [Figure 2-8](#) and described in [Table 2-3](#).

Figure 2-8. Channel Options Parameter (OPT)

| | | | | | | | | | | | | |
|------|----------|--------|--------|---------|--------|----------|---------|----------|---------|-------|-------|-------|
| 31 | 30 | 28 | 27 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| PRIV | Reserved | | PRIVID | ITCCHEN | TCCHEN | ITCINTEN | TCINTEN | Reserved | | | | TCC |
| R-0 | R-0 | | R-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | | | R/W-0 | | R/W-0 |
| 15 | 12 | 11 | 10 | 8 | 7 | | 4 | 3 | 2 | 1 | 0 | |
| | TCC | TCCMOD | FWID | | | Reserved | | STATIC | SYNCDIM | DAM | SAM | |
| | R/W-0 | R/W-0 | R/W-0 | | | R/W-0 | | R/W-0 | R/W-0 | R/W-0 | R/W-0 | |

Table 2-3. Channel Options Parameters (OPT) Field Descriptions

| Bit | Field | Value | Description |
|-------|----------|--------|---|
| 31 | PRIV | 0 1 | Privilege level (supervisor versus user) for the host/CPU/DMA that programmed this PaRAM set. This value is set with the EDMA3 master's privilege value when any part of the PaRAM set is written. User level privilege Supervisor level privilege |
| 30-28 | Reserved | 0 | Reserved |
| 27-24 | PRIVID | 0-Fh | Privilege identification for the external host/CPU/DMA that programmed this PaRAM set. This value is set with the EDMA3 master's privilege identification value when any part of the PaRAM set is written. |
| 23 | ITCCHEN | 0 1 | Intermediate transfer completion chaining enable. 0 Intermediate transfer complete chaining is disabled 1 Intermediate transfer complete chaining is enabled When enabled, the chained event register (CER/CERH) bit is set on every intermediate chained transfer completion (upon completion of every intermediate TR in the PaRAM set, except the final TR in the PaRAM set). The bit (position) set in CER or CERH is the TCC value specified. |
| 22 | TCCHEN | 0 1 | Transfer complete chaining enable. 0 Transfer complete chaining is disabled 1 Transfer complete chaining is enabled When enabled, the chained event register (CER/CERH) bit is set on final chained transfer completion (upon completion of the final/last TR in the PaRAM set). The bit (position) set in CER or CERH is the TCC value specified. |
| 21 | ITCINTEN | 0 1 | Intermediate transfer completion interrupt enable. 0 Intermediate transfer complete interrupt is disabled 1 Intermediate transfer complete interrupt is enabled When enabled, the interrupt pending register (IPR/IPRH) bit is set on every intermediate transfer completion (upon completion of every intermediate TR in the PaRAM set, except the final TR in the PaRAM set). The bit (position) set in IPR or IPRH is the TCC value specified. To generate a completion interrupt to the CPU, the corresponding IER[TCC]/IERH[TCC] bit must be set. |
| 20 | TCINTEN | 0 1 | Transfer complete interrupt enable. 0 Transfer complete interrupt is disabled 1 Transfer complete interrupt is enabled When enabled, the interrupt pending register (IPR/IPRH) bit is set on transfer completion (upon completion of the final TR in the PaRAM set). The bit (position) set in IPR or IPRH is the TCC value specified. To generate a completion interrupt to the CPU, the corresponding IER[TCC]/IERH[TCC] bit must be set. |
| 19 | Reserved | 0 | Reserved. Always write 0 to this field. |
| 18 | Reserved | 0 | Reserved |
| 17-12 | TCC | 0-3Fh | Transfer complete code. This 6-bit code sets the relevant bit in the chaining enable register (CER[TCC]/CERH[TCC]) for chaining or in the interrupt pending register (IPR[TCC]/IPRH[TCC]) for interrupts. |

Table 2-3. Channel Options Parameters (OPT) Field Descriptions (continued)

| Bit | Field | Value | Description |
|------|----------|-------|--|
| 11 | TCCMODE | | Transfer complete code mode. Indicates the point at which a transfer is considered completed for chaining and interrupt generation. |
| | | 0 | Normal completion: A transfer is considered completed after the data has been transferred. |
| | | 1 | Early completion: A transfer is considered completed after the EDMA3CC submits a TR to the EDMA3TC. TC may still be transferring data when the interrupt/chain is triggered. |
| 10-8 | FWID | 0-7h | FIFO Width. Applies if either SAM or DAM is set to FIFO mode. |
| | | 0 | FIFO width is 8-bit |
| | | 1h | FIFO width is 16-bit |
| | | 2h | FIFO width is 32-bit |
| | | 3h | FIFO width is 64-bit |
| | | 4h | FIFO width is 128-bit |
| | | 5h | FIFO width is 256-bit |
| | | 6h-7h | Reserved |
| 7-4 | Reserved | 0 | Reserved |
| 3 | STATIC | | Static set. |
| | | 0 | Set is not static. The PaRAM set is updated or linked after a TR is submitted. A value of 0 should be used for DMA channels and for nonfinal transfers in a linked list of QDMA transfers. |
| | | 1 | Set is static. The PaRAM set is not updated or linked after a TR is submitted. A value of 1 should be used for isolated QDMA transfers or for the final transfer in a linked list of QDMA transfers. |
| 2 | SYNCDIM | | Transfer synchronization dimension. |
| | | 0 | A-synchronized. Each event triggers the transfer of a single array of ACNT bytes. |
| | | 1 | AB-synchronized. Each event triggers the transfer of BCNT arrays of ACNT bytes. |
| 1 | DAM | | Destination address mode. |
| | | 0 | Increment (INCR) mode. Destination addressing within an array increments. Destination is not a FIFO. |
| | | 1 | FIFO mode. Destination addressing within an array wraps around upon reaching FIFO width. |
| 0 | SAM | | Source address mode. |
| | | 0 | Increment (INCR) mode. Source addressing within an array increments. Source is not a FIFO. |
| | | 1 | FIFO mode. Source addressing within an array wraps around upon reaching FIFO width. |

2.3.2.2 Channel Source Address (SRC)

The 32-bit source address parameter specifies the starting byte address of the source. For SAM in increment mode, there are no alignment restrictions imposed by EDMA3. For SAM in FIFO mode, you must program the source address to be aligned to a 256-bit aligned address (the 5 LSBs of the address must be 0). The EDMA3TC signals an error if this rule is violated; for additional details, see [Section 2.12.3](#).

2.3.2.3 Channel Destination Address (DST)

The 32-bit destination address parameter specifies the starting byte address of the destination. For DAM in increment mode, there are no alignment restrictions imposed by EDMA3. For DAM in FIFO mode, you must program the destination address to be aligned to a 256-bit aligned address (the 5 LSBs of the address must be 0). The EDMA3TC signals an error if this rule is violated; for additional details, see [Section 2.12.3](#).

2.3.2.4 Count for 1st Dimension (ACNT)

ACNT represents the number of bytes within the 1st dimension of a transfer. ACNT is a 16-bit unsigned value with valid values between 0 and 65 535. Therefore, the maximum number of bytes in an array is 65 535 bytes (64K - 1 bytes). ACNT must be greater than or equal to 1 for a TR to be submitted to EDMA3TC. An 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 OPT.

For details on dummy/null completion conditions, see [Section 2.3.5](#) and [Section 2.5.3](#).

2.3.2.5 Count for 2nd Dimension (BCNT)

BCNT is a 16-bit unsigned value that specifies the number of arrays of length ACNT. For normal operation, valid values for BCNT are between 1 and 65 535. Therefore, the maximum number of arrays in a frame is 65 535 (64K - 1 arrays). A 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 OPT.

For details on dummy/null completion conditions, see [Section 2.3.5](#) and [Section 2.5.3](#).

2.3.2.6 Count for 3rd Dimension (CCNT)

CCNT is a 16-bit unsigned value that specifies the number of frames in a block. Valid values for CCNT are between 1 and 65 535. Therefore, the maximum number of frames in a block is 65 535 (64K - 1 frames). A 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 OPT.

A CCNT value of 0 is considered either a null or dummy transfer. For details on dummy/null completion conditions, see [Section 2.3.5](#) and [Section 2.5.3](#).

2.3.2.7 BCNT Reload (BCNTRLD)

BCNTRLD is a 16-bit unsigned value used to reload the BCNT field once the last array in the 2nd dimension is transferred. This field is only used for A-synchronized transfers. In this case, the EDMA3CC decrements the BCNT value by 1 on each TR submission. When BCNT (conceptually) reaches 0, the EDMA3CC decrements CCNT and uses the BCNTRLD value to reinitialize the BCNT value.

For AB-synchronized transfers, the EDMA3CC submits the BCNT in the TR and the EDMA3TC decrements BCNT appropriately. For AB-synchronized transfers, BCNTRLD is not used.

2.3.2.8 Source B Index (SRCBIDX)

SRCBIDX is a 16-bit signed value (2s complement) used for source address modification between each array in the 2nd dimension. Valid values for SRCBIDX are between -32 768 and 32 767. It provides a byte address offset from the beginning of the source array to the beginning of the next source array. It applies to both A-synchronized and AB-synchronized transfers. Some examples:

- SRCBIDX = 0000h (0): no address offset from the beginning of an array to the beginning of the next array. All arrays are fixed to the same beginning address.
- SRCBIDX = 0003h (+3): the address offset from the beginning of an array to the beginning of the next array in a frame is 3 bytes. For example, if the current array begins at address 1000h, the next array begins at 1003h.
- SRCBIDX = FFFFh (-1): the address offset from the beginning of an array to the beginning of the next array in a frame is -1 byte. For example, if the current array begins at address 5054h, the next array begins at 5053h.

2.3.2.9 Destination B Index (DSTBIDX)

DSTBIDX is a 16-bit signed value (2s complement) used for destination address modification between each array in the 2nd dimension. Valid values for DSTBIDX are between -32 768 and 32 767. It provides a byte address offset from the beginning of the destination array to the beginning of the next destination array within the current frame. It applies to both A-synchronized and AB-synchronized transfers. For examples, see [Section 2.3.2.8](#).

2.3.2.10 Source C Index (SRCCIDX)

SRCCIDX is a 16-bit signed value (2s complement) used for source address modification in the 3rd dimension. Valid values for SRCCIDX are between -32 768 and 32 767. It provides a byte address offset from the beginning of the current array (pointed to by SRC address) to the beginning of the first source array in the next frame. It applies to both A-synchronized and AB-synchronized transfers. When SRCCIDX is applied, the current array in an A-synchronized transfer is the last array in the frame (Figure 2-5), while the current array in an AB-synchronized transfer is the first array in the frame (Figure 2-6).

2.3.2.11 Destination C Index (DSTCIDX)

DSTCIDX is a 16-bit signed value (2s complement) used for destination address modification in the 3rd dimension. Valid values are between -32 768 and 32 767. It provides a byte address offset from the beginning of the current array (pointed to by DST address) to the beginning of the first destination array TR in the next frame. It applies to both A-synchronized and AB-synchronized transfers. When DSTCIDX is applied, the current array in an A-synchronized transfer is the last array in the frame (Figure 2-5), while the current array in a AB-synchronized transfer is the first array in the frame (Figure 2-6).

2.3.2.12 Link Address (LINK)

The EDMA3CC provides a mechanism, called linking, to reload the current PaRAM set upon its natural termination (that is, after the count fields are decremented to 0) with a new PaRAM set. The 16-bit parameter LINK specifies the byte address offset in the PaRAM from which the EDMA3CC loads/reloads the next PaRAM set during linking.

You must program the link address to point to a valid aligned 32-byte PaRAM set. The 5 LSBs of the LINK field should be cleared to 0.

The EDMA3CC ignores the upper two bits of the LINK entry, allowing the programmer the flexibility of programming the link address as either an absolute/literal byte address or use the PaRAM-base-relative offset address. Therefore, if you make use of the literal address with a range from 4000h to 7FFFh, it is treated as a PaRAM-base-relative value of 0000h to 3FFFh.

You should make sure to program the LINK field correctly, so that link update is requested from a PaRAM address that falls in the range of the available PaRAM addresses on the device.

A LINK value of FFFFh is referred to as a NULL link that should cause the EDMA3CC to perform an internal write of 0 to all entries of the current PaRAM set, except for the LINK field that is set to FFFFh. For details on terminating a transfer, see Section 2.5.

2.3.3 Null PaRAM Set

A null PaRAM set is defined as a PaRAM set where all count fields (ACNT, BCNT, and CCNT) are cleared to 0. If a PaRAM set associated with a channel is a NULL set, then when serviced by the EDMA3CC, the bit corresponding to the channel is set in the associated event missed register (EMR, EMRH, or QEMR). This bit remains set in the associated secondary event register (SER, SERH, or QSER). This implies that any future events on the same channel are ignored by the EDMA3CC and you are required to clear the bit in SER, SERH, or QSER for the channel. This is considered an error condition, since events are not expected on a channel that is configured as a null transfer. For more information on the SER and EMR registers, respectively, see Section 4.2.6.8 and Section 4.2.2.1.

2.3.4 Dummy PaRAM Set

A dummy PaRAM set is defined as a PaRAM set where at least one of the count fields (ACNT, BCNT, or CCNT) is cleared to 0 and at least one of the count fields is nonzero.

If a PaRAM set associated with a channel is a dummy set, then when serviced by the EDMA3CC, it does not set the bit corresponding to the channel (DMA/QDMA) in the event missed register (EMR, EMRH, or QEMR) and the secondary event register (SER, SERH, or QSER) bit gets cleared similar to a normal transfer. Future events on that channel are serviced. A dummy transfer is a legal transfer of 0 bytes. For more information on the SER and EMR registers, respectively, see Section 4.2.6.8 and Section 4.2.2.1.

2.3.5 Dummy Versus Null Transfer Comparison

There are some differences in the way the EDMA3CC logic treats a dummy versus a null transfer request. A null transfer request is an error condition, but a dummy transfer is a legal transfer of 0 bytes. A null transfer causes an error bit (En) in EMR to get set and the En bit in SER remains set, essentially preventing any further transfers on that channel without clearing the associated error registers.

[Table 2-4](#) summarizes the conditions and effects of null and dummy transfer requests.

Table 2-4. Dummy and Null Transfer Request

| Feature | Null TR | Dummy TR |
|---|---------|----------|
| EMR/EMRH/QEMR is set | Yes | No |
| SER/SERH/QSER remains set | Yes | No |
| Link update (STATIC = 0 in OPT) | Yes | Yes |
| QER is set | Yes | Yes |
| IPR/IPRH CER/CERH is set using early completion | Yes | Yes |

2.3.6 Parameter Set Updates

When a TR is submitted for a given DMA/QDMA channel and its corresponding PaRAM set, the EDMA3CC is responsible for updating the PaRAM set in anticipation of the next trigger event. For nonfinal events, this includes address and count updates; for final events, this includes the link update.

The specific PaRAM set entries that are updated depend on the channel's synchronization type (A-synchronized or B-synchronized) and the current state of the PaRAM set. A B-update refers to the decrementing of BCNT in the case of A-synchronized transfers after the submission of successive TRs. A C-update refers to the decrementing of CCNT in the case of A-synchronized transfers after BCNT TRs for ACNT byte transfers have submitted. For AB-synchronized transfers, a C-update refers to the decrementing of CCNT after submission of every transfer request.

For details and conditions on the parameter updates, see [Table 2-5](#). A link update occurs when the PaRAM set is exhausted, as described in [Section 2.3.7](#).

After the TR is read from the PaRAM (and is in process of being submitted to EDMA3TC), the following fields are updated if needed:

- A-synchronized: BCNT, CCNT, SRC, DST
- AB-synchronized: CCNT, SRC, DST

The following fields are not updated (except for during linking, where all fields are overwritten by the link PaRAM set):

- A-synchronized: ACNT, BCNTRLD, SRCBIDX, DSTBIDX, SRCCIDX, DSTCIDX, OPT, LINK
- AB-synchronized: ACNT, BCNT, BCNTRLD, SRCBIDX, DSTBIDX, SRCCIDX, DSTCIDX, OPT, LINK

Note that PaRAM updates only pertain to the information that is needed to properly submit the next transfer request to the EDMA3TC. Updates that occur while data is moved within a transfer request is tracked within the transfer controller, and is detailed in [Section 2.12](#). For A-synchronized transfers, the EDMA3CC always submits a TRP for ACNT bytes (BCNT = 1 and CCNT = 1). For AB-synchronized transfers, the EDMA3CC always submits a TRP for ACNT bytes of BCNT arrays (CCNT = 1). The EDMA3TC is responsible for updating source and destination addresses within the array based on ACNT and FWID (in OPT). For AB-synchronized transfers, the EDMA3TC is also responsible to update source and destination addresses between arrays based on SRCBIDX and DSTBIDX.

[Table 2-5](#) shows the details of parameter updates that occur within EDMA3CC for A-synchronized and AB-synchronized transfers.

Table 2-5. Parameter Updates in EDMA3CC (for Non-Null, Non-Dummy PaRAM Set)

| Condition: | A-Synchronized Transfer | | | AB-Synchronized Transfer | | |
|--------------------|-------------------------|--------------------------|---------------------------|--------------------------|------------|----------------|
| | B-Update | C-Update | Link Update | B-Update | C-Update | Link Update |
| | BCNT > 1 | BCNT == 1 && CCNT > 1 | BCNT == 1 && CCNT == 1 | N/A | CCNT > 1 | CCNT == 1 |
| SRC | += SRCBIDX | += SRCCIDX | = Link.SRC | in EDMA3TC | += SRCCIDX | = Link.SRC |
| DST | += DSTBIDX | += DSTCIDX | = Link.DST | in EDMA3TC | += DSTCIDX | = Link.DST |
| ACNT | None | None | = Link.ACNT | None | None | = Link.ACNT |
| BCNT | -= 1 | = BCNTRLD | = Link.BCNT | in EDMA3TC | N/A | = Link.BCNT |
| CCNT | None | -= 1 | = Link.CCNT | in EDMA3TC | -=1 | = Link.CCNT |
| SRCBIDX | None | None | = Link.SRCBIDX | in EDMA3TC | None | = Link.SRCBIDX |
| DSTBIDX | None | None | = Link.DSTBIDX | None | None | = Link.DSTBIDX |
| SRCCIDX | None | None | = Link.SRCBIDX | in EDMA3TC | None | = Link.SRCBIDX |
| DSTCIDX | None | None | = Link.DSTBIDX | None | None | = Link.DSTBIDX |
| LINK | None | None | = Link.LINK | None | None | = Link.LINK |
| BCNTRLD | None | None | = Link.BCNTRLD | None | None | = Link.BCNTRLD |
| OPT ⁽¹⁾ | None | None | = LINK.OPT | None | None | = LINK.OPT |

⁽¹⁾ In all cases, no updates occur if OPT.STATIC == 1 for the current PaRAM set.

Note: The EDMA3CC includes no special hardware to detect when an indexed address update calculation overflows/underflows. The address update wraps across boundaries as programmed. You should ensure that no transfer is allowed to cross internal port boundaries between peripherals. A single TR must target a single source/destination slave endpoint.

2.3.7 Linking Transfers

The EDMA3CC provides a mechanism known as linking, which allows the entire PaRAM set to be reloaded from a location within the PaRAM memory map (for both DMA and QDMA channels). Linking is especially useful for maintaining ping-pong buffers, circular buffering, and repetitive/continuous transfers with no CPU intervention. Upon completion of a transfer, the current transfer parameters are reloaded with the parameter set pointed to by the 16-bit link address field of the current parameter set. Linking only occurs when the STATIC bit in OPT is cleared.

Note: A transfer (EDMA3 or QDMA) should always be linked to another useful transfer. If a transfer must be terminated, the transfer should be linked to a NULL parameter set (see [Section 2.3.3](#)).

The link update occurs after the current PaRAM set event parameters have been exhausted. An event's parameters are exhausted when the EDMA3 channel controller has submitted all the transfers associated with the PaRAM set.

A link update occurs for null and dummy transfers depending on the state of the STATIC bit in OPT and the LINK field. In both cases (null or dummy), if the value of LINK is FFFFh, then a null PaRAM set (with all 0s and LINK set to FFFFh) is written to the current PaRAM set. Similarly, if LINK is set to a value other than FFFFh then the appropriate PaRAM location pointed to by LINK is copied to the current PaRAM set.

Once the channel completion conditions are met for an event, the transfer parameters located at the link address are loaded into the current DMA or QDMA channel's associated parameter set. This indicates that the EDMA3CC reads the entire set (8 words) from the PaRAM set specified by LINK and writes all 8 words to the PaRAM set associated with the current channel. [Figure 2-9](#) shows an example of a linked EDMA3CC transfer.

Any PaRAM set in the PaRAM can be used as a link/reload parameter set.

If a PaRAM set location is defined as a QDMA channel PaRAM set (by QCHMAP n), then copying the link PaRAM set into the current QDMA channel PaRAM set is recognized as a trigger event. It is latched in QER because a write to the trigger word was performed. This feature can be used to create a linked list of transfers using a single QDMA channel and multiple PaRAM sets (see [Section 2.4.2](#)).

Linking to itself replicates the behavior of autoinitialization, which facilitates the use of circular buffering and repetitive transfers. After an EDMA3 channel exhausts its current PaRAM set, it reloads the parameter set and the transfer begins again.

Note: If the STATIC bit in OPT is set for a PaRAM set, then link updates are not performed.

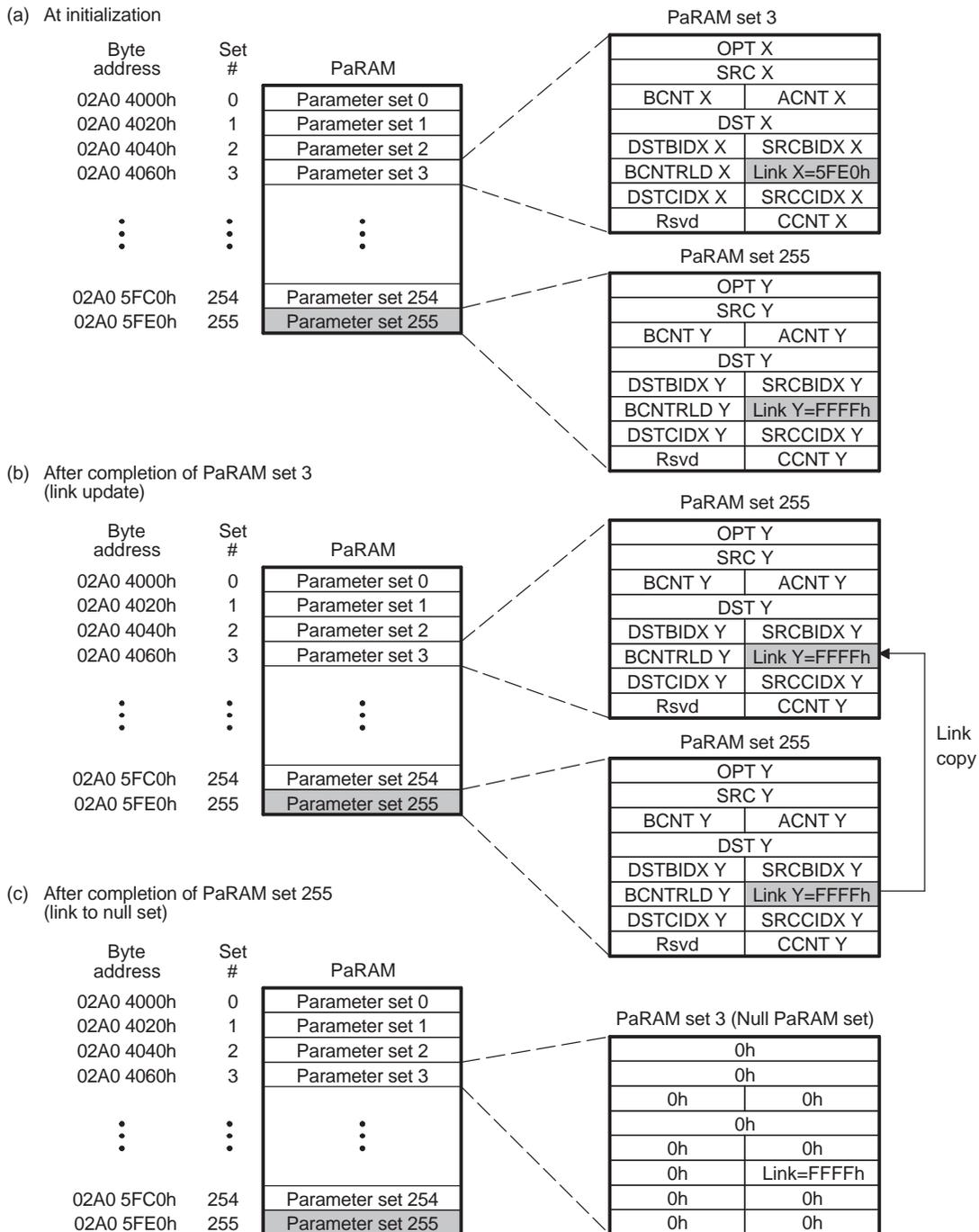
2.3.8 Fixed-Mode Transfers/Alignment Issues

If either SAM or DAM is set (FIFO mode), then the source or destination address must be aligned to a 256-bit aligned address, respectively, and the corresponding BIDX should be an even multiple of 32 bytes (256 bits). The EDMA3CC does not recognize errors here but the EDMA3TC asserts an error if this is not true (see [Section 2.12.3](#)).

2.3.9 Element Size

The EDMA3 controller does not use element-size and element-indexing. Instead, all transfers are defined in terms of all three dimensions: ACNT, BCNT, and CCNT. An element-indexed transfer is logically achieved by programming ACNT to the size of the element and BCNT to the number of elements that need to be transferred. For example, if you have 16-bit audio data and 256 audio samples that must be transferred to a serial port, this can be done by programming the ACNT = 2 (2 bytes) and BCNT = 256.

Figure 2-9. Linked Transfer



2.4 Initiating a DMA Transfer

There are multiple ways to initiate a programmed data transfer using the EDMA3 channel controller. Transfers on DMA channels are initiated by three sources:

- **Event-triggered transfer request** (this is the more typical usage of EDMA3): A peripheral, system, or externally-generated event triggers a transfer request.
- **Manually-triggered transfer request:** The CPU to manually triggers a transfer by writing a 1 to the corresponding bit in the event set register (ESR/ESRH).
- **Chain-triggered transfer request:** A transfer is triggered on the completion of another transfer or subtransfer.

Transfers on QDMA channels are initiated by two sources:

- Autotriggered transfer request: A transfer is triggered when the programmed trigger word is written to.
- Link-triggered transfer requests: When linking occurs, the transfer is triggered as soon as the trigger word is written to.

2.4.1 DMA Channel

2.4.1.1 Event-Triggered Transfer Request

When an event is asserted from a peripheral or device pins, it gets latched in the corresponding bit of the event register ($ER.En = 1$). For peripheral event-to-DMA event mapping, see [Table 2-6](#). If the corresponding event in the event enable register (EER) is enabled ($EER.En = 1$), then the EDMA3CC prioritizes and queues the event in the appropriate event queue. When the event reaches the head of the queue, it is evaluated for submission as a transfer request to the transfer controller.

If the PaPARAM set is valid (not a NULL set), then a transfer request packet (TRP) is submitted to the EDMA3TC and the En bit in ER is cleared. At this point, a new event can be safely received by the EDMA3CC.

If the PaPARAM set associated with the channel is a NULL set (see [Section 2.3.3](#)), then no transfer request (TR) is submitted and the corresponding En bit in ER is cleared and simultaneously the corresponding channel bit is set in the event miss register ($EMR.En = 1$) to indicate that the event was discarded due to a null TR being serviced. Good programming practices should include cleaning the event missed error before retriggering the DMA channel.

When an event is received, the corresponding event bit in the event register is set ($ER.En = 1$), regardless of the state of $EER.En$. If the event is disabled when an external event is received ($ER.En = 1$ and $EER.En = 0$), the $ER.En$ bit remains set. If the event is subsequently enabled ($EER.En = 1$), then the pending event is processed by the EDMA3CC and the TR is processed/submitted, after which the $ER.En$ bit is cleared.

If an event is being processed (prioritized or is in the event queue) and another sync event is received for the same channel prior to the original being cleared ($ER.En \neq 0$), then the second event is registered as a missed event in the corresponding bit of the event missed register ($EMR.En = 1$).

Table 2-6. EDMA3 Channel Synchronization Events

| EDMA3 Channel | Event Name | Event Description |
|------------------|-------------|---|
| 0 ⁽¹⁾ | DSP_EVT | HPI/PCI-to-DSP Event |
| 1 | TEVTLO0 | Timer 0 Lower Counter Event |
| 2 | TEVTHI0 | Timer 0 Higher Counter Event |
| 3-8 | - | None |
| 9 | ETBHFULLINT | Embedded Trace Buffer (ETB) is Half Full |
| 10 | ETBFULLINT | Embedded Trace Buffer (ETB) is Full |
| 11 | ETBACQINT | Embedded Trace Buffer (ETB) Acquisition is Complete |

⁽¹⁾ HPI boot is terminated using a DSP interrupt. The DSP interrupt is registered in bit 0 (channel 0) of the EDMA Event Register (ER). This event must be cleared by software before triggering transfers on DMA channel 0.

Table 2-6. EDMA3 Channel Synchronization Events (continued)

| EDMA3 Channel | Event Name | Event Description |
|---------------|------------|-----------------------------|
| 12 | XEVT0 | MCBSP0 Transmit Event |
| 13 | REVT0 | MCBSP0 Receive Event |
| 14 | XEVT1 | MCBSP1 Transmit Event |
| 15 | REVT1 | MCBSP1 Receive Event |
| 16 | TINT1L | Timer1 Lower Counter Event |
| 17 | TINT1H | Timer1 Higher Counter Event |
| 18 | - | None |
| 19 | INTDST0 | RapidIO Interrupt 0 |
| 20 | INTDST1 | RapidIO Interrupt 1 |
| 21 | INTDST2 | RapidIO Interrupt 2 |
| 22 | INTDST3 | RapidIO Interrupt 3 |
| 23 | INTDST4 | RapidIO Interrupt 4 |
| 24 | INTDST5 | RapidIO Interrupt 5 |
| 25 | INTDST6 | RapidIO Interrupt 6 |
| 26-27 | - | None |
| 28 | VCPREVT | VCP2 Receive Event |
| 29 | VCPXEVT | VCP2 Transmit Event |
| 30 | TCP2AREVT | TCP2_A Receive Event |
| 31 | TCP2AXEVT | TCP2_A Transmit Event |
| 32 | UREVT | UTOPIA Receive Event |
| 33 | TCP2BREVT | TCP2_B Receive Event |
| 34 | TCP2BXEVT | TCP2_B Transmit Event |
| 35-39 | - | None |
| 40 | UXEVT | UTOPIA Transmit Event |
| 41-43 | - | None |
| 44 | ICREVT | I2C Receive Event |
| 45 | ICXEVT | I2C Transmit Event |
| 46-47 | - | None |
| 48 | GPINT0 | GPIO Event 0 |
| 49 | GPINT1 | GPIO Event 1 |
| 50 | GPINT2 | GPIO Event 2 |
| 51 | GPINT3 | GPIO Event 3 |
| 52 | GPINT4 | GPIO Event 4 |
| 53 | GPINT5 | GPIO Event 5 |
| 54 | GPINT6 | GPIO Event 6 |
| 55 | GPINT7 | GPIO Event 7 |
| 56 | GPINT8 | GPIO Event 8 |
| 57 | GPINT9 | GPIO Event 9 |
| 58 | GPINT10 | GPIO Event 10 |
| 59 | GPINT11 | GPIO Event 11 |
| 60 | GPINT12 | GPIO Event 12 |
| 61 | GPINT13 | GPIO Event 13 |
| 62 | GPINT14 | GPIO Event 14 |
| 63 | GPINT15 | GPIO Event 15 |

2.4.1.2 Manually-Triggered Transfer Request

A DMA transfer is initiated by a write to the event set register (ESR) by the CPU (or any EDMA programmer). Writing a 1 to an event bit in the ESR results in the event being prioritized/queued in the appropriate event queue, regardless of the state of the EER.En bit. When the event reaches the head of the queue, it is evaluated for submission as a transfer request to the transfer controller.

As in the event-triggered transfers, if the PaRAM set associated with the channel is valid (it is not a null set) then the TR is submitted to the associated EDMA3TC and the channel can be triggered again.

If the PaRAM set associated with the channel is a NULL set (see [Section 2.3.3](#)), then no transfer request (TR) is submitted and the corresponding En bit in ER is cleared and simultaneously the corresponding channel bit is set in the event miss register (EMR.En = 1) to indicate that the event was discarded due to a null TR being serviced. Good programming practices should include clearing the event missed error before retriggering the DMA channel.

If an event is being processed (prioritized or is in the event queue) and the same channel is manually set by a write to the corresponding channel bit of the event set register (ESR.En = 1) prior to the original being cleared (ESR.En = 0), then the second event is registered as a missed event in the corresponding bit of the event missed register (EMR.En = 1).

2.4.1.3 Chain-Triggered Transfer Request

Chaining is a mechanism by which the completion of one transfer automatically sets the event for another channel. When a chained completion code is detected, the value of which is dictated by the transfer completion code (TCC[5:0] in OPT of the PaRAM set associated with the channel), it results in the corresponding bit in the chained event register (CER) to be set (CER.E[TCC] = 1).

Once a bit is set in CER, the EDMA3CC prioritizes and queues the event in the appropriate event queue. When the event reaches the head of the queue, it is evaluated for submission as a transfer request to the transfer controller.

As in the event-triggered transfers, if the PaRAM set associated with the channel is valid (it is not a null set) then the TR is submitted to the associated EDMA3TC and the channel can be triggered again.

If the PaRAM set associated with the channel is a NULL set (see [Section 2.3.3](#)), then no transfer request (TR) is submitted and the corresponding En bit in CER is cleared and simultaneously the corresponding channel bit is set in the event miss register (EMR.En = 1) to indicate that the event was discarded due to a null TR being serviced. In this case, the error condition must be cleared by you before the DMA channel can be retriggered. Good programming practices might include clearing the event missed error before retriggering the DMA channel.

If a chaining event is being processed (prioritized or queued) and another chained event is received for the same channel prior to the original being cleared (CER.En != 0), then the second chained event is registered as a missed event in the corresponding channel bit of the event missed register (EMR.En = 1).

Note: Chained event registers, event registers, and event set registers operate independently. An event (En) can be triggered by any of the trigger sources (event-triggered, manually-triggered, or chain-triggered).

2.4.2 QDMA Channels

2.4.2.1 Autotriggered and Link-Triggered Transfer Request

QDMA-based transfer requests are issued when a QDMA event gets latched in the QDMA event register (QER.En = 1). A bit corresponding to a QDMA channel is set in the QDMA event register (QER) when the following occurs:

- A CPU (or any EDMA3 programmer) write address occurs to a PaRAM address that is defined as a QDMA channel trigger word (programmed in the QDMA channel mapping register (QCHMAP n)) for the particular QDMA channel and the QDMA channel is enabled via the QDMA event enable register (QEER.En = 1).
- EDMA3CC performs a link update on a PaRAM set address that is configured as a QDMA channel (matches QCHMAP n settings) and the corresponding channel is enabled via the QDMA event enable register (QEER.En = 1).

Once a bit is set in QER, the EDMA3CC prioritizes and queues the event in the appropriate event queue. When the event reaches the head of the queue, it is evaluated for submission as a transfer request to the transfer controller.

As in the event-triggered transfers, if the PaPARAM set associated with the channel is valid (it is not a null set) then the TR is submitted to the associated EDMA3TC and the channel can be triggered again.

If a bit is already set in QER (QER.En = 1) and a second QDMA event for the same QDMA channel occurs prior to the original being cleared, the second QDMA event gets captured in the QDMA event miss register (QEMR.En = 1).

2.4.3 Comparison Between DMA and QDMA Channels

The primary difference between DMA and QDMA channels is the event/channel synchronization. QDMA events are either autotriggered or link triggered. Autotriggering allows QDMA channels to be triggered by CPU(s) with a minimum number of linear writes to PaPARAM. Link triggering allows a linked list of transfers to be executed, using a single QDMA PaPARAM set and multiple link PaPARAM sets.

A QDMA transfer is triggered when a CPU (or other EDMA3 programmer) writes to the trigger word of the QDMA channel parameter set (autotriggered) or when the EDMA3CC performs a link update on a PaPARAM set that has been mapped to a QDMA channel (link triggered). Note that for CPU triggered (manually triggered) DMA channels, in addition to writing to the PaPARAM set, it is required to write to the event set register (ESR) to kick-off the transfer.

QDMA channels are typically for cases where a single event accomplishes a complete transfer since the CPU (or EDMA3 programmer) must reprogram some portion of the QDMA PaPARAM set in order to retrigger the channel. In other words, QDMA transfers are programmed with BCNT = CCNT = 1 for A-synchronized transfers, and CCNT = 1 for AB-synchronized transfers.

Additionally, since linking is also supported (if STATIC = 0 in OPT) for QDMA transfers, it allows you to initiate a linked list of QDMAs, so when EDMA3CC copies over a link PaPARAM set (including the write to the trigger word), the current PaPARAM set mapped to the QDMA channel is automatically recognized as a valid QDMA event and initiate another set of transfers as specified by the linked set.

2.5 Completion of a DMA Transfer

The TCC is used to generate an interrupt to the CPU and/or chain trigger another channel. Details on the variations of this feature are described later.

For details on interrupts, see [Section 2.9](#) and for details on chaining, see [Section 2.8](#).

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 2-7](#) for both synchronization types along with state of the PaPARAM set prior to the final TR being submitted. When the counts (BCNT and/or CCNT) are this value, the next TR results in a:

- Final chaining or interrupt codes to be sent by the transfer controllers (instead of intermediate).

- Link updates (linking to either null or another valid link set).

Table 2-7. Expected Number of Transfers for Non-Null Transfer

| Sync Mode | Counts at time 0 | Total # Transfers | Counts prior to final TR |
|-----------------|----------------------|---------------------------------------|--------------------------|
| A-synchronized | ACNT BCNT CCNT | (BCNT × CCNT) TRs of ACNT bytes each | BCNT == 1 && CCNT == 1 |
| AB-synchronized | ACNT BCNT CCNT | CCNT TRs for ACNT × BCNT bytes each | CCNT == 1 |

You must program the PaRAM OPT field with a specific transfer completion code (TCC) along with the other OPT fields 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 (TCCHEN, TCINTEN, ITCCHEN, and ITCINTEN bits in OPT).

The specific TCC value (6-bit binary value) programmed dictates which of the 64-bits in the chain event register (CER[TCC]) and/or interrupt pending register (IPR[TCC]) is set.

You can also selectively program whether the transfer controller sends back completion codes on completion of final transfer request (TR) of a parameter set (TCCHEN or TCINTEN), for all but the final transfer request (TR) of a parameter set (ITCCHEN or ITCINTEN), or for all TRs of a parameter set (both). For details on chaining (intermediate/final chaining), see [Section 2.8](#) and for details on intermediate/final interrupt completion, see [Section 2.9](#).

A completion detection interface exists between the EDMA3 channel controller and transfer controller(s). This interface sends back information from the transfer controller to the channel controller to indicate that a specific transfer is completed. 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 should point to another predefined PaRAM set. Alternatively, a nonrepetitive transfer should set the link address value to the null link value. The null link value is defined as FFFFh. For more details, see [Section 2.3.7](#).

Note: Any incoming events that are mapped to a null PaRAM set results in an error condition. The error condition should be cleared before the corresponding channel is used again (see [Section 2.3.5](#)).

There are three ways the EDMA3CC gets updated/informed about a transfer completion: normal completion, early completion, and dummy/null completion. This applies to both chained events and completion interrupt generation.

2.5.1 Normal Completion

In normal completion mode (TCCMODE = 0 in OPT), the transfer or sub-transfer is considered to be complete when the EDMA3 channel controller receives the completion codes from the EDMA3 transfer controller. In this mode, the completion code to the channel controller is posted by the transfer controller after it receives a signal from the destination peripheral. Normal completion is typically used to generate an interrupt to inform the CPU that a set of data is ready for processing.

2.5.2 Early Completion

In early completion mode (TCCMODE = 1 in OPT), the transfer is considered to be complete when the EDMA3 channel controller submits the transfer request (TR) to the EDMA3 transfer controller. In this mode, the channel controller generates the completion code internally. Early completion is typically useful for chaining, as it allows subsequent transfers to be chained-triggered while the previous transfer is still in progress within the transfer controller, maximizing the overall throughput of the set of the transfers.

2.5.3 Dummy or Null Completion

This is a variation of early completion. Dummy or null completion is associated with a dummy set ([Section 2.3.4](#)) or null set ([Section 2.3.3](#)). In both cases, the EDMA3 channel controller does not submit the associated transfer request to the EDMA3 transfer controller(s). However, if the set (dummy/null) has the OPT field programmed to return completion code (intermediate/final interrupt/chaining completion), then it sets the appropriate bits in the interrupt pending registers (IPR/IPRH) or chained event register (CER/CERH). The internal early completion path is used by the channel controller to return the completion codes internally (that is, EDMA3CC generates the completion code).

2.6 Event, Channel, and PaRAM Mapping

Several of the 64 DMA channels are tied to a specific hardware event, thus allowing transfers to be triggered by events from device peripherals or external hardware. A DMA channel typically requests a data transfer when it receives its event (apart from manually-triggered, chain-triggered, and other transfers). The amount of data transfer per synchronization event depends on the channel's configuration (ACNT, BCNT, CCNT, etc.) and the synchronization type (A-synchronized or AB-synchronized).

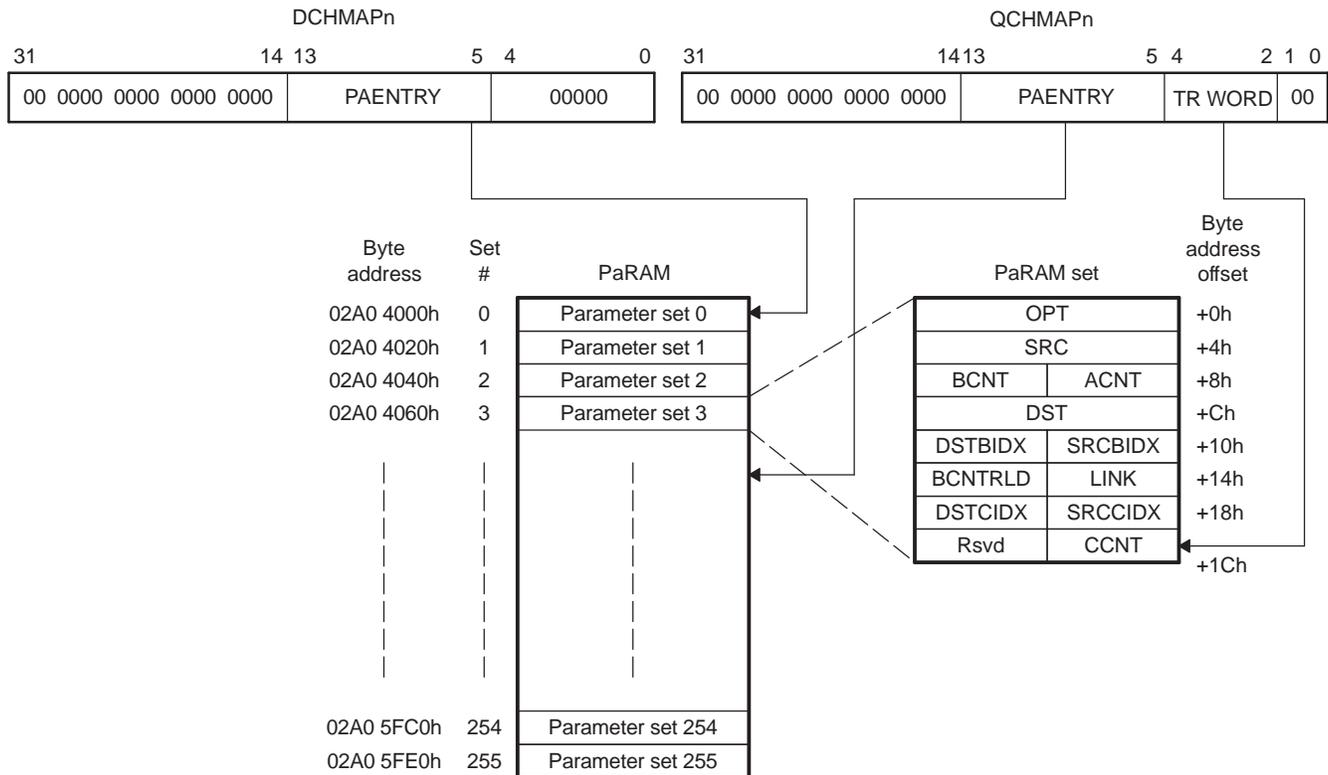
The association of an event to a channel is fixed, each DMA channel has one specific event associated with it. [Table 2-6](#) provides the synchronization events associated with each of the programmable DMA channels.

In an application, if a channel does not use the associated synchronization event or does not have an associated synchronization event (unused), the PaRAM set associated with that channel can be used for manually-triggered or chained-triggered transfers, for linking/reloading, or as a QDMA channel.

2.6.1 DMA Channel to PaRAM Mapping

The mapping between the DMA channel numbers and the PaRAM sets is programmable (see [Table 2-1](#)). The DMA channel mapping registers (DCHMAP_n) in the EDMA3CC provide programmability that allows the DMA channels to be mapped to any of the PaRAM sets in the PaRAM memory map. [Figure 2-10](#) illustrates the use of DCHMAP. There is one DCHMAP register per channel. For more information, see [Table 2-1](#).

Figure 2-10. DMA Channel and QDMA Channel to PaRAM Mapping



2.6.2 QDMA Channel to PaRAM Mapping

The mapping between the QDMA channels and the PaRAM sets is programmable. The QDMA channel mapping register (QCHMAP) in the EDMA3CC allows the QDMA channels to be mapped to any of the PaRAM sets in the PaRAM memory map. [Figure 2-10](#) illustrates the use of QCHMAP.

Additionally, QCHMAP allows you to program the trigger word in the PaRAM set for the QDMA channel. A trigger word is one of the 8 words in the PaRAM set. For a QDMA transfer to occur, a valid TR synchronization event for EDMA3CC is a write to the trigger word in the PaRAM set pointed to by QCHMAP for a particular QDMA channel. By default, QDMA channels are mapped to PaRAM set 0. PaRAM set 0 must be appropriately remapped before it is used.

2.7 EDMA3 Channel Controller Regions

The EDMA3 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 EDMA programmer.

Application software can be designed to use regions or ignore them altogether. Active memory protection can be used in conjunction with regions such that only a specific EDMA programmer (for example, privilege identification) or privilege level (for example, user versus 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 programmer only modifies the state of the assigned resources. Memory protection is described in [Section 2.10](#).

2.7.1 Region Overview

The EDMA3 channel controller memory-mapped registers are divided in three main regions:

1. Global region
2. Global channel region
3. Shadow region(s)

The global region is located at a single/fixed location in the EDMA3CC memory map. Registers in the global region control EDMA3 resource mapping and provide debug visibility and error tracking information. For the EDMA3CC memory map, see the *TMS320C6457 Fixed-Point Digital Signal Processor* data manual ([SPRS582](#)).

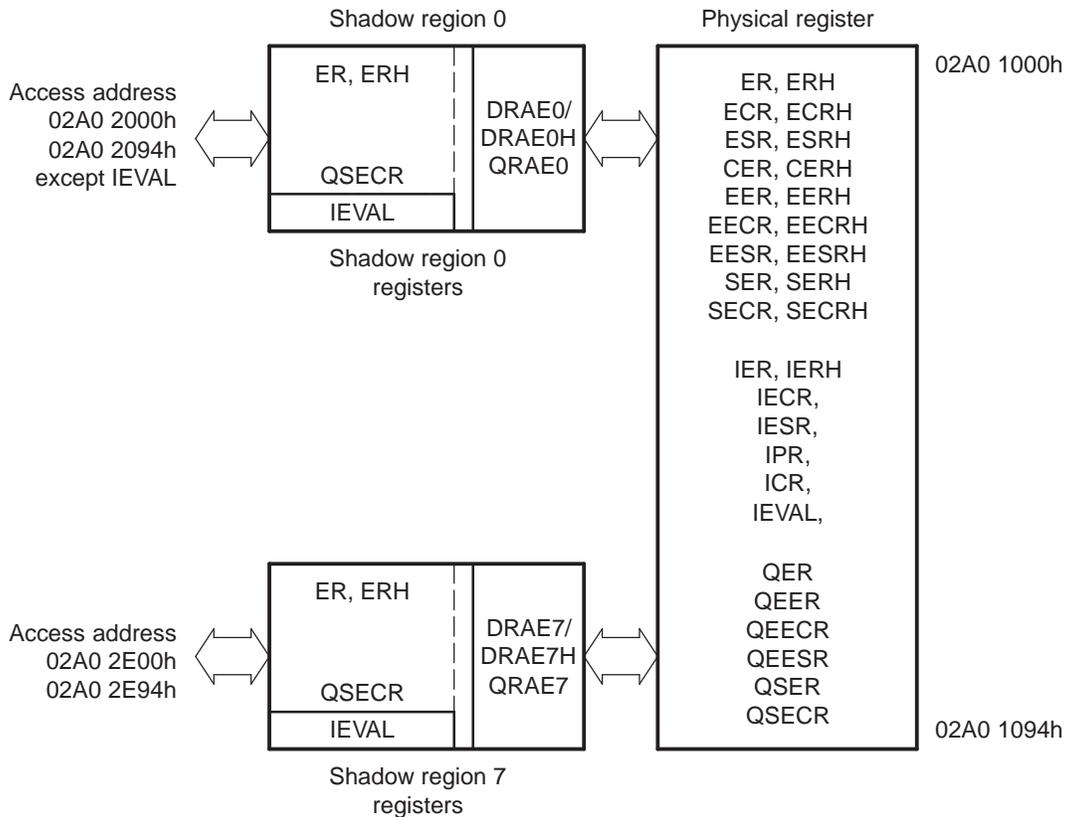
The channel registers (including DMA, QDMA, and interrupt registers) are accessible via the global channel region address range, or in the shadow n channel region address range(s). For example, the event enable register (EER) is visible at the global address of 02A0 1020h or region addresses of 02A0 2020h for region 0, 02A0 2220h for region 1, ... 02A0 2E20h for region 7.

The underlying control register bits that are accessible via the shadow region address space (except for IEVAL n) are controlled by the DMA region access enable registers (DRAE m) and QDMA region access enable registers (QRAE n). [Table 2-8](#) lists the registers in the shadow region memory map. For the complete global and shadow region memory maps, see the EDMA3CC memory map ([Table 4-1](#)). [Figure 2-11](#) illustrates the conceptual view of the regions.

Table 2-8. Shadow Region Registers

| DRAE _m | DRAEH _m | QRAE _n |
|--|--------------------|-------------------|
| ER | ERH | QER |
| ECR | ECRH | QEER |
| ESR | ESRH | QEESR |
| CER | CERH | QEESR |
| EER | EERH | |
| EECR | EECRH | |
| EESR | EESRH | |
| SER | SERH | |
| SECR | SECRH | |
| IER | IERH | |
| IECR | IECRH | |
| IESR | IESRH | |
| IPR | IPRH | |
| ICR | ICRH | |
| Register not affected by DRAE\DRAEH | | |
| IEVAL | | |

Figure 2-11. Shadow Region Registers



2.7.2 Channel Controller Regions

There are eight EDMA3 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.

- **DRAEm** and **DRAEHm**: One register pair exists for each of the shadow regions. The number of bits in each register matches the number of DMA channels (64 DMA channels). These registers need to be programmed to assign ownership of DMA channels and interrupt channels (or TCC codes) to the respective region. Accesses to DMA and interrupt registers via the shadow region address view are filtered through the DRAE/DRAEH pair. A value of 1 in the corresponding DRAE(H) bit implies that the corresponding DMA/interrupt channel is accessible; a value of 0 in the corresponding DRAE(H) bit forces writes to be discarded and returns a value of 0 for reads.
- **QRAEn**: One register exists for every region. The number of bits in each register matches the number of QDMA channels (8 QDMA channels). These registers must be programmed to assign ownership of QDMA channels to the respective region. To enable a channel in a shadow region using shadow region 0 QEER, the respective bit in QRAE must be set or writing into QEESR does not have the desired effect.
- **MPPAn** and **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.

Therefore, the use of shadow regions allows access to EDMA3 resources (DMA channels, QDMA channels, TCC, interrupts) by tasks in a system in a mutually-exclusive way. The mutual exclusiveness is not assured by the EDMA3 hardware and would need to be implemented in software.

Example 2-1. Resource Pool Division Across Two Regions

This example illustrates a judicious resource pool division across two regions, assuming region 0 needs to 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 3 QDMA channels (1-3) and TCC codes (16-47). DRAE should be equal to the OR of the bits required for the DMA channels and the TCC codes:

```
Region 0:
    DRAEH, DRAE = 0xFFFF0000, 0x0000FFFF
    QRAE = 0x0000001

Region 1:
    DRAEH, DRAE = 0x0000FFFF, 0xFFFF0000
    QRAE = 0x000000E
```

2.7.3 Region Interrupts

In addition to the EDMA3CC global completion interrupt, there is an additional completion interrupt line associated with every shadow region. Along with the interrupt enable register (IER), DRAE acts as a secondary interrupt enable for the respective shadow region interrupts. For more information, see [Section 2.9](#).

2.8 Chaining EDMA3 Channels

The channel chaining capability for the EDMA3 allows the completion of an EDMA3 channel transfer to trigger another EDMA3 channel transfer. The purpose is to allow you the ability to chain several events through one event occurrence.

Chaining is different from linking ([Section 2.3.7](#)). The EDMA3 link feature reloads the current channel parameter set with the linked parameter set. The EDMA3 chaining feature does not modify or update any channel parameter set; it provides a synchronization event to the chained channel. For chain-triggered transfer requests, see [Section 2.4.1.3](#).

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 TCC bit of channel m channel options parameter (OPT) set.

- If final transfer completion chaining (TCCHEN = 1 in OPT) 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 (ITCCHEN = 1 in OPT) 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 (TCCHEN = 1 and ITCCHEN = 1 in OPT) are enabled, then the chain-trigger event occurs after every transfer request is submitted or completed (depending on early or normal completion).

[Table 2-9](#) illustrates the number of chain event triggers occurring in different synchronized scenarios. Consider channel 31 programmed with ACNT = 3, BCNT = 4, CCNT = 5, and TCC = 30.

Table 2-9. Chain Event Triggers

| | (Number of chained event triggers on channel 30) | |
|-------------------------|--|----------------------------------|
| | A-Synchronized | AB-Synchronized |
| TCCHEN = 1, ITCCHEN = 0 | 1 (Owing to the last TR) | 1 (Owing to the last TR) |
| TCCHEN = 0, ITCCHEN = 1 | 59 (Owing to all but the last TR) | 4 (Owing to all but the last TR) |
| TCCHEN = 1, ITCCHEN = 1 | 60 (Owing to a total of 60 TRs) | 5 (Owing to a total of 5 TRs) |

2.9 EDMA3 Interrupts

The EDMA3 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 2-10](#) and the error interrupts are listed in [Table 2-11](#). The transfer completion interrupts and the error interrupts from the transfer controllers are all routed directly to DSP interrupt controller.

Table 2-10. EDMA3 Transfer Completion Interrupts

| Name | Description | DSP Interrupt Number |
|--------------|---|----------------------|
| EDMA3CC_ | EDMA3CC Channel Global Completion Interrupt | 24 |
| EDMA3CC_INT0 | EDMA3CC Transfer Completion Interrupt - Mask0 | 71 |
| EDMA3CC_INT1 | EDMA3CC Transfer Completion Interrupt - Mask1 | 72 |
| EDMA3CC_INT2 | EDMA3CC Transfer Completion Interrupt - Mask2 | 73 |
| EDMA3CC_INT3 | EDMA3CC Transfer Completion Interrupt - Mask3 | 74 |
| EDMA3CC_INT4 | EDMA3CC Transfer Completion Interrupt - Mask4 | 75 |
| EDMA3CC_INT5 | EDMA3CC Transfer Completion Interrupt - Mask5 | 76 |
| EDMA3CC_INT6 | EDMA3CC Transfer Completion Interrupt - Mask6 | 77 |
| EDMA3CC_INT7 | EDMA3CC Transfer Completion Interrupt - Mask7 | 78 |
| EDMA3CC_AET | EDMA3CC AET Event | 85 |

Table 2-11. EDMA3 Error Interrupts

| Name | Description | Event Number |
|-----------------|-------------------------------------|--------------|
| EDMA3CC_ERRINT | EDMA3CC Error Interrupt | |
| EDMA3CC_MPINT | EDMA3CC Memory Protection Interrupt | |
| EDMA3TC0_ERRINT | EDMA3TC0 Error Interrupt | 81 |
| EDMA3TC1_ERRINT | EDMA3TC1 Error Interrupt | 82 |
| EDMA3TC2_ERRINT | EDMA3TC2 Error Interrupt | 83 |
| EDMA3TC3_ERRINT | EDMA3TC3 Error Interrupt | 84 |
| EDMA3TC4_ERRINT | EDMA3TC4 Error Interrupt | 86 |
| EDMA3TC5_ERRINT | EDMA3TC5 Error Interrupt | 87 |

Note: Different error interrupts are routed to DSP cores according to the Chip Interrupt Controller (CIC) logic. This is programmable to the user. For more details, see the *TMS320C6457 Fixed-Point Digital Signal Processor* data manual ([SPRS582](#)).

2.9.1 Transfer Completion Interrupts

The EDMA3CC is responsible for generating transfer completion interrupts to the CPU(s) (and other EDMA3 masters). The EDMA3 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 EDMA3 interrupt generation.

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

The transfer completion code (TCC) value is directly mapped to the bits of the interrupt pending register (IPR/IPRH). For example, if TCC = 10 0001b, 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. For details on enabling EDMA3 transfer completion interrupts, see [Section 2.9.1.1](#).

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

The transfer completion code (TCC) can be programmed to any value for a DMA/QDMA channel. There does not need to be a direct relation between the channel number and the transfer completion code value. This allows multiple channels having the same transfer completion code value to cause a CPU to execute the same interrupt service routine (ISR) for different channels.

If the channel is used in the context of a shadow region and you intend for the shadow region interrupt to be asserted, then ensure that the bit corresponding to the TCC code is enabled in IER/IERH and in the corresponding shadow region's DMA region access registers (DRAE/DRAEH).

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 (TCCINT = 1 in OPT) is enabled, the interrupt occurs after the last transfer request of channel m is either submitted or completed (depending on early or normal completion).
- If the intermediate transfer interrupt (ITCCINT = 1 in OPT) 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 (TCCINT = 1, and ITCCINT = 1 in OPT) are enabled, then the interrupt occurs after every transfer request is submitted or completed (depending on early or normal completion).

Table 2-12 shows the number of interrupts occurring in different synchronized scenarios. Consider channel 31 programmed with ACNT = 3, BCNT = 4, CCNT = 5, and TCC = 30.

Table 2-12. Number of Interrupts

| Options | A-Synchronized | AB-Synchronized |
|-------------------------|-----------------------------------|----------------------------------|
| TCCINT = 1, ITCCINT = 0 | 1 (Owing to the last TR) | 1 (Owing to the last TR) |
| TCCINT = 0, ITCCINT = 1 | 59 (Owing to all but the last TR) | 4 (Owing to all but the last TR) |
| TCCINT = 1, ITCCINT = 1 | 60 (Owing to a total of 60 TRs) | 5 (Owing to a total of 5 TRs) |

Note: Because the DMA region access registers (DRAE/DRAEH) are required to enable the transfer completion region interrupts, it is assumed that there is a unique and nonoverlapping (in most cases) assignment of the channels and interrupts among the different shadow regions. This allows the interrupt registers (IER, IESR, IECR, IPR, and ICR) in the different shadow regions to functionally operate in an independent and nonoverlapping manner. The interrupt service routine examples are based on this assumption.

2.9.1.1 Enabling Transfer Completion Interrupts

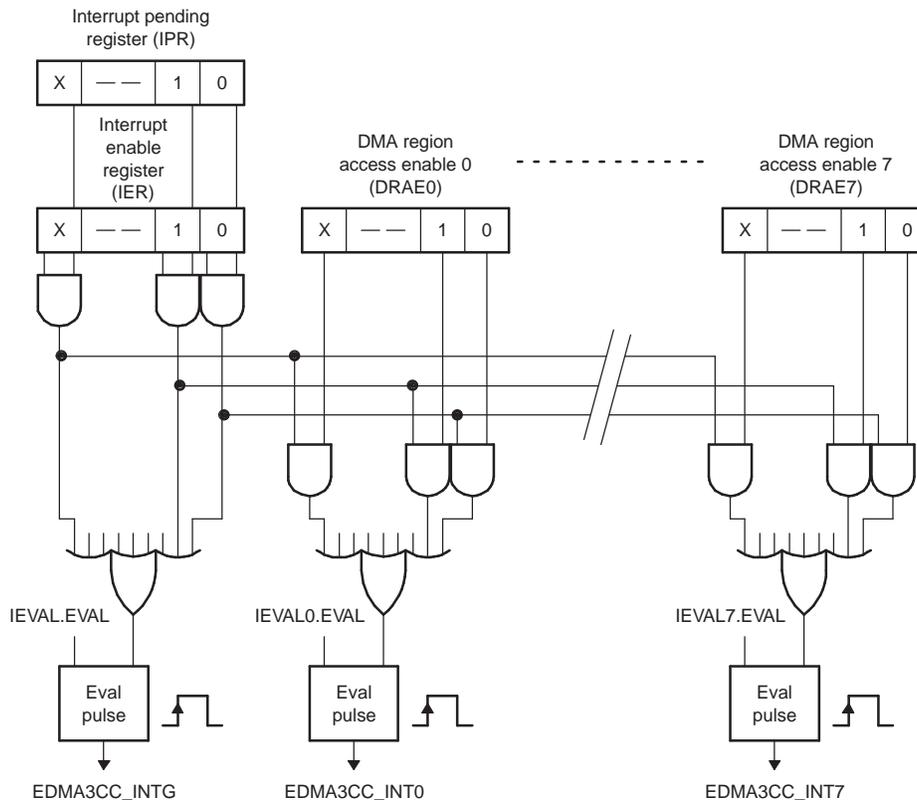
For the EDMA3 channel controller to assert a transfer completion to the external environment, the interrupts have to be enabled in the EDMA3CC. This is in addition to setting up the TCINTEN and ITCINTEN bits in OPT of the associated PaRAM set.

The EDMA3 channel controller has interrupt enable registers (IER/IERH) and each bit location in IER/IERH serves as a primary enable for the corresponding interrupt pending registers (IPR/IPRH).

All the interrupt registers (IER, IESR, IECR, and 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 EDMA3 channel controller has a hierarchical completion interrupt scheme that uses a single set of interrupt pending registers (IPR/IPRH) and single set of interrupt enable registers (IER/IERH). A second level of interrupt masking is provided by the programmable DMA region access enable registers (DRAE/DRAEH). The global region interrupt output is gated based on the enable mask provided by IER. The region interrupt outputs are gated by IER and the specific DRAE/DRAEH associated with the region (see Figure 2-12).

Figure 2-12. Interrupt Diagram



For the EDMA3CC to generate the transfer completion interrupts associated with each shadow region, the following conditions must be true:

- EDMA3CC_INT0: (IPR.E0 & IER.E0 & DRAE0.E0) | (IPR.E1 & IER.E1 & DRAE0.E1) | ...|(IPRH.E63 & IERH.E63 & DRAHE0.E63)
- EDMA3CC_INT1: (IPR.E0 & IER.E0 & DRAE1.E0) | (IPR.E1 & IER.E1 & DRAE1.E1) | ...| (IPRH.E63 & IERH.E63 & DRAHE1.E63)
- EDMA3CC_INT2 : (IPR.E0 & IER.E0 & DRAE2.E0) | (IPR.E1 & IER.E1 & DRAE2.E1) | ...|(IPRH.E63 & IERH.E63 & DRAHE2.E63)....
- Up to EDMA3CC_INT7 : (IPR.E0 & IER.E0 & DRAE7.E0) | (IPR.E1 & IER.E1 & DRAE7.E1) | ...|(IPRH.E63 & IERH.E63 & DRAEH7.E63)

Note: The DRAE/DRAEH for all regions are expected to be set up at system initialization and to remain static for an extended period of time. It is the interrupt enable registers should be used for dynamic enable/disable of individual interrupts.

Because there is no relation between the TCC value and the DMA/QDMA channel, it is possible, for example, for DMA channel 0 to have the OPT.TCC = 63 in its associated PaPARAM set. This would mean that if a transfer completion interrupt is enabled (OPT.TCINTEN or OPT.ITCINTEN is set), then based on the TCC value, IPRH.E63 is set on completion. For proper channel operations and interrupt generation using the shadow region map, the DRAE/DRAEH associated with the shadow region must be programmed to have read/write access to both bit 0 (corresponding to channel 0) and bit 63 (corresponding to IPRH bit that is set on completion).

2.9.1.2 Clearing Transfer Completion Interrupts

Transfer completion interrupts that are latched to the interrupt pending registers (IPR/IPRH) are cleared by writing a 1 to the corresponding bit in the interrupt pending clear register (ICR/ICRH). For example, a write of 1 to ICR.E0 clears a pending interrupt in IPR.E0.

If an incoming transfer completion code (TCC) gets latched to a bit in IPR/IPRH, then additional bits that get set due to subsequent transfer completion do not result in asserting the EDMA3CC completion interrupt. In order for the completion interrupt to be pulsed, the required transition is from a state of no enabled interrupts are set to at least one enabled interrupt is set.

2.9.2 EDMA3 Interrupt Servicing

On completion of a transfer (early or normal completion), the EDMA3 channel controller sets the appropriate bit in the interrupt pending registers (IPR/IPRH) as specified by the transfer completion codes. If the completion interrupts are appropriately enabled, then the CPU enters the interrupt service routine (ISR) when the completion interrupt is asserted.

After servicing the interrupt, the ISR should clear the corresponding bit in IPR/IPRH, thereby enabling recognition of future interrupts. Only when all IPR/IPRH bits are cleared, the EDMA3CC asserts additional completion interrupts.

When one interrupt is serviced many other transfer completions may result in additional bits being set in IPR/IPRH, thereby resulting in additional interrupts. It is likely that each of these bits in IPR/IPRH would need different types of service; therefore, the ISR may check all pending interrupts and continue until all the posted interrupts are appropriately serviced.

Following are examples (pseudo code) for a CPU interrupt service routine for an EDMA3CC completion interrupt.

The ISR routine in [Example 2-2](#) is more exhaustive and incurs a higher latency. [Example 2-3](#) is less rigorous, with less burden on the software in polling for set interrupt bits, but can occasionally (although rarely) cause a race condition as mentioned above.

Example 2-2. Interrupt Servicing

The pseudo code:

1. Reads the interrupt pending register (IPR/IPRH).
2. Performs the operations needed.
3. Writes to the interrupt pending clear register (ICR/ICRH) to clear the corresponding IPR/IPRH bit(s).
4. Reads IPR/IPRH again:
 - a. If IPR/IPRH is not equal to 0, repeat from step 2 (implies occurrence of new event between step 2 to step 4).
 - b. If IPR/IPRH is equal to 0, this should assure you that all enabled interrupts are inactive.

Note: An event may occur during step 4 while the IPR/IPRH bits are read to be 0 and the application is still in the interrupt service routine. If this happens, a new interrupt is recorded in the device interrupt controller and a new interrupt is generated as soon as the application exits the interrupt service routine.

Example 2-3. Interrupt Servicing

If it is desired to leave any enabled and pending (possibly lower priority) interrupts; it is required to force the interrupt logic to reassert the interrupt pulse. To do so, you must reenable the interrupt by setting the bit in the interrupt evaluation register (IEVAL).

The pseudo code:

1. Enters ISR. (The corresponding bit (EDMA3CC_INT* bit) in the DSP IFR is cleared).
2. Reads IPR/IPRH.
3. For the condition set in IPR/IPRH that you desire to service:
 - a. Service interrupt as required by application.
 - b. Clear bit for serviced conditions (others may still be set, and other transfers may have resulted in returning the TCC to EDMA3CC after step 2).
4. Reads IPR/IPRH prior to exiting ISR:
 - a. If IPR/IPRH is equal to 0, then exit ISR.
 - b. If IPR/IPRH is not equal to 0, then set IEVAL so that upon exit of ISR, a new interrupt is triggered if any enabled interrupts are still pending.

Conditions 4a and 4b must be differentiated to avoid extra pulses on the interrupt output signal. If the IEVAL bit is set when IPR/IPRH = 0, there is a chance that a new transfer completion code is latched to IPR/IPRH before the write to the EVAL bit. Therefore, EDMA3CC generates an completion interrupt both when the enabled interrupt is set (IPR/IPRH.En = 1) and when the write to IEVAL completes. This should be avoided in software.

2.9.3 Interrupt Evaluation Operations

The EDMA3CC has interrupt evaluate registers (IEVAL) that exist in both the global region and in each shadow region. These 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 (DRAE/DRAEH). A write of 1 to the EVAL bit in these registers associated with a particular shadow region results in pulsing the associated region interrupt (global or shadow), if any enabled interrupt (via IER/IERH) is still pending (IPR/IPRH). This register assures that the interrupts are not missed by the CPU (or the EDMA3 master associated with the shadow region) if the software architecture chooses not to use all interrupts. For the use of IEVAL in the EDMA3 interrupt service routine (ISR), see [Example 2-3](#).

Similarly an error evaluation register (EEVAL) exists in the global region. A write of 1 to the EVAL bit in EEVAL causes the pulsing of the error interrupt if any pending errors are in EMR/EMRH, QEMR, or CCERR. For additional details on error interrupts, see [Section 2.9.4](#).

Note: While using IEVAL for shadow region completion interrupts, you should make sure that the IEVAL operated upon is from that particular shadow region memory map.

2.9.4 Error Interrupts

The EDMA3CC error registers provide the capability to differentiate error conditions (event missed, threshold exceed, etc.). Additionally, if the error bits are set in these registers, it results in asserting the EDMA3CC error interrupt. If EDMA3CC error interrupt is enabled in the device interrupt controller(s), then it allows the CPU(s) to handle the error conditions.

The EDMA3CC has a single error interrupt (EDMA3CC_ERRINT) that gets asserted for all EDMA3CC error conditions. There are four conditions that cause the error interrupt to be pulsed:

- DMA missed events: for all 64 DMA channels. These get latched in the event missed registers (EMR/EMRH).
- QDMA missed events: for all QDMA channels. These get latched in the QDMA event missed register (QEMR).
- Threshold exceed: for all event queues. These get latched in EDMA3CC error register (CCERR).
- TCC error: for outstanding transfer requests expected to return completion code (TCCHEN or TCINTEN bit in OPT is set to 1) exceeding the maximum limit of 63. This also gets latched in the EDMA3CC error register (CCERR).

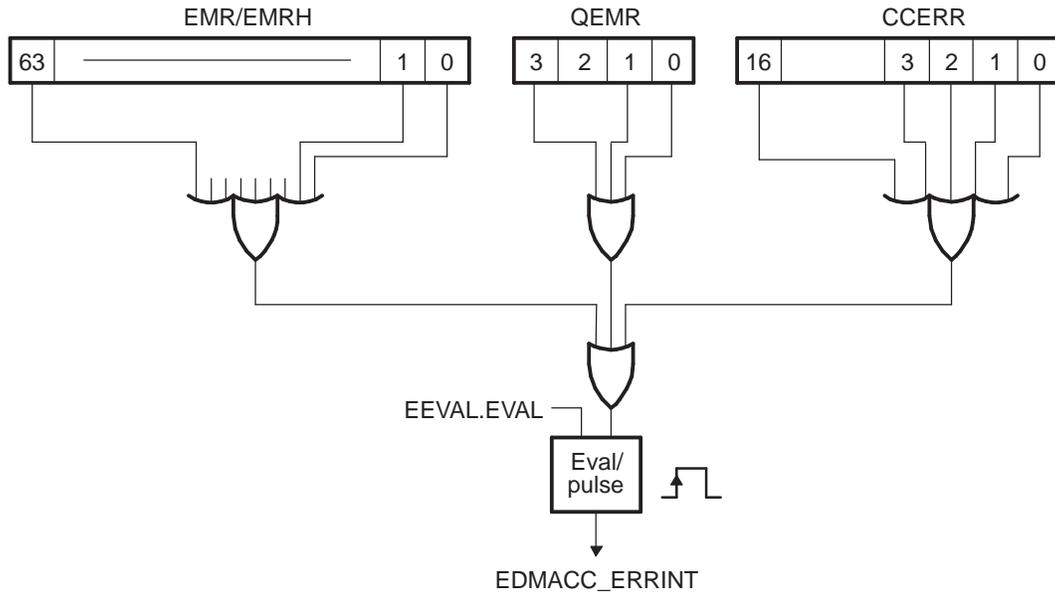
[Figure 2-13](#) illustrates the EDMA3CC error interrupt generation operation.

If any of the bits are set in the error registers due to any error condition, the EDMA3CC_ERRINT always is asserted, as there are no enables for masking these error events. Similar to transfer completion interrupts (EDMA3CC_INT*), the error interrupt also is pulsed only when the error interrupt condition transitions from a no errors are set to at least one error is set. If additional error events are latched prior to the original error bits being cleared, the EDMA3CC does not generate additional interrupt pulses.

To reduce the burden on the software, similar to the interrupt evaluate register (IEVAL), there is an error evaluate register (EEVAL) that allows reevaluation of pending set error events/bits. This can be used so that the CPU(s) does not miss any error events.

Note: It is a good practice to have the error interrupt enabled in the device interrupt controller and associate an interrupt service routine with it to address the various error conditions appropriately. This puts less burden on software (polling for error status) and additionally provides a good debug mechanism for unexpected error conditions.

Figure 2-13. Error Interrupt Operation



2.10 Memory Protection

There are two kinds of memory protection supported by the EDMA3 channel controller: active and proxied.

2.10.1 Active Memory Protection

Active memory protection allows read/write accesses (by any EDMA programmer) to the EDMA3CC registers to be accepted/denied based on user-programmed permission characteristics. This is achieved by a set of memory protect permissions attribute (MPPA) registers. The EDMA3CC register map is divided into a global region and eight shadow regions. All eight regions have a respective MPPA register (MPPA n) that defines the specific requestor(s) and types of requests that are allowed to the region's memory-mapped registers as well as the region's PaRAM section. The global region is protected with MPPAG. The MPPAG applies to all the EDMA3CC registers except the PaRAM, DMA shadow region, and MPPA registers themselves. For the bit field descriptions of MPPA n , see [Section 4.2.5.4](#). The MPPA n have a certain set of access rules. [Table 2-13](#) shows the accesses that are allowed or disallowed to the MPPAG and MPPA n .

Table 2-13. Allowed Accesses

| Access | Supervisor | User |
|--------|------------|------|
| Read | Yes | Yes |
| Write | Yes | No |

[Table 2-14](#) describes the MPPA register mapping for the shadow regions (which includes shadow region registers and PaRAM addresses).

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

The first example in [Figure 2-14](#) illustrates a user write access to shadow region 7 EESR when the MPPA7 has user write accesses allowed. The second example in [Figure 2-14](#) illustrates a user write access to shadow region 7 EESR when the MPPA7 does not allow user write accesses.

Table 2-14. MPPA Registers to Region Assignment

| Register | Registers Protect | Address Range | PaRAM Protect | Address Range |
|----------|-------------------|---------------|---------------|---------------|
| MPPAG | Global Range | 0000h-1FFCh | NA | NA |
| MPPA0 | DMA Shadow 0 | 2000h-21FCh | 1st octant | 4000h-43FCh |
| MPPA1 | DMA Shadow 1 | 2200h-23FCh | 2nd octant | 4400h-47FCh |
| MPPA2 | DMA Shadow 2 | 2400h-25FCh | 3rd octant | 4800h-4BFCh |
| MPPA3 | DMA Shadow 3 | 2600h-27FCh | 4th octant | 4C00h-4FFCh |
| MPPA4 | DMA Shadow 4 | 2800h-29FCh | 5th octant | 5000h-53FCh |
| MPPA5 | DMA Shadow 5 | 2A00h-2BFCh | 6th octant | 5400h-57FCh |
| MPPA6 | DMA Shadow 6 | 2C00h-2DFCh | 7th octant | 5800h-5BFCh |
| MPPA7 | DMA Shadow 7 | 2E00h-2FFCh | 8th octant | 5C00h-5FFCh |

2.10.2 Proxied Memory Protection

Proxied memory protection allows a DMA transfer programmed by a given CPU to have its permissions travel with the transfer through the TR and also through the EDMA3TC for the read and write transfers to the source and destination endpoints. The PRIV bit (supervisor versus user) and PRIVID bit in the channel options parameter (OPT) is set with the EDMA programmer's priv value and privid values, respectively, when any part of the PaRAM set is written. For the PRIVIDs associated with potential EDMA programmers, see the *TMS320C6457 Fixed-Point Digital Signal Processor* data manual ([SPRS582](#)).

These options are part of the TR 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 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 PRIV is 0 for user level and the CPU has a privid of 0. The PaRAM set is shown in [Figure 2-15](#).

Figure 2-15. PaRAM Set Content for Proxied Memory Protection Example

(a) EDMA Parameters

| Parameter Contents | | Parameter | |
|--------------------|-------|-----------------------------------|--------------------------------|
| 0010 0007h | | Channel Options Parameter (OPT) | |
| 1088 0000h | | Channel Source Address (SRC) | |
| 0001h | 0004h | Count for 2nd Dimension (BCNT) | Count for 1st Dimension (ACNT) |
| 10F0 7800h | | Channel Destination Address (DST) | |
| 0001h | 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 | 0001h | Reserved | Count for 3rd Dimension (CCNT) |

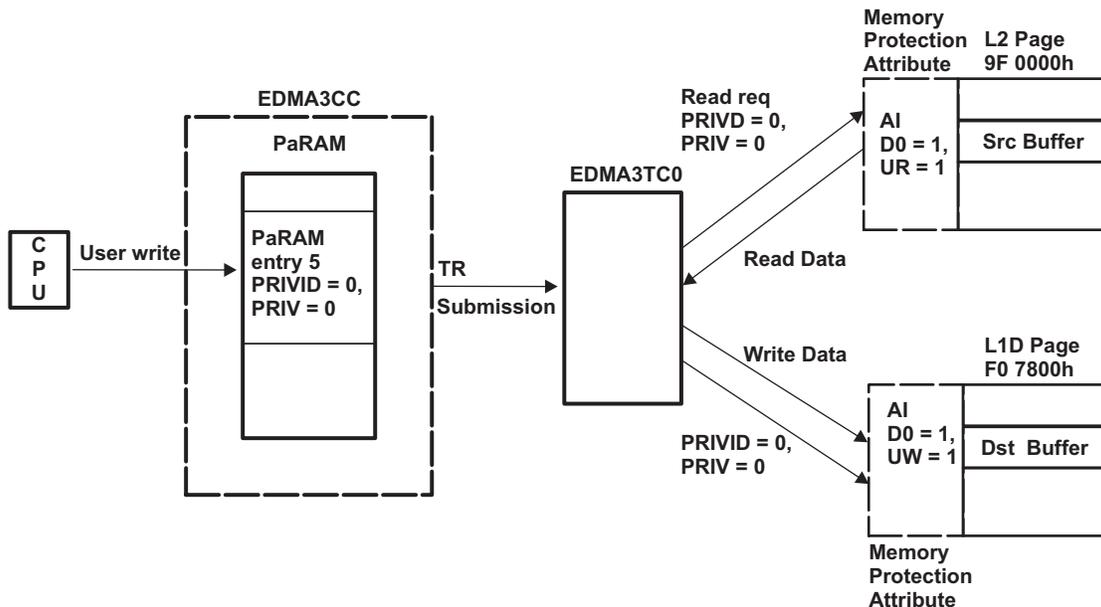
(b) Channel Options Parameter (OPT) Content

| | | | | | | | | | | | | | |
|------|------|--------|--------|------|---------|----------|----------|---------|----------|----|---------|-----|-----|
| 31 | 30 | 29 | 28 | 27 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| 0 | 0 | 00 | 0000 | | 0 | 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 |

The PRIV and PRIVID information travels along with the read and write requests issued to the source and destination memories.

For example, if the access attributes associated with the L2 page with the source buffer allow only supervisor read, write accesses (SR, SW), the above user level read request would be refused. Similarly, if the access attributes associated with the L1D page with the destination buffer allow only supervisor read, write accesses (SR, SW), the above user level write request would be refused. For the transfer to succeed, the source and destination pages should have user read and user write permissions, respectively, along with allowing accesses from a PRIVID 0. Because the programmers privilege level and privilege identification travel with the read and write requests, EDMA3 acts as a proxy. [Figure 2-16](#) illustrates the propagation of PRIV and PRIVID at the boundaries of all the interacting entities (CPU, EDMA3CC, EDMA3TC, and slave memories).

Figure 2-16. Proxied Memory Protection Example



2.11 Event Queue(s)

Event queues are a part of the EDMA3 channel controller. Event queues form the interface between the event detection logic in the EDMA3CC and the transfer request (TR) submission logic of the EDMA3CC. Each queue is 16 entries deep; thus, a maximum of 16 events can be queued per event queue. If there are more than 16 events, then the events that cannot find a place in the event queue remain set in the associated event register and the CPU is not stalled.

There are six event queues (Queue0, Queue1, Queue2, Queue3, Queue 4 and Queue5) for TMS320C6457 devices. Events in Queue0 result in submission of its associated transfer requests (TRs) to TC0. Similarly, transfer requests associated with events in Queue5 are submitted to TC5.

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 a FIFO (first in-first out) order. Once the event reaches the head of its queue and the corresponding transfer controller is ready to receive another TR, the event is dequeued and the PaRAM set corresponding to the dequeued event is processed and submitted as a transfer request packet (TRP) to the associated EDMA3 transfer controller.

Queue0 has highest priority and Queue5 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 dequeued first and its associated PaRAM set is processed and submitted as a transfer request (TR) to TC0.

For system-level performance considerations, see [Section 2.11.4](#). All the event entries in all the event queues are software readable (not writeable) by accessing the event entry registers (Q0E0, Q0E1,...Q1E15, etc.). Each event entry register characterizes the queued event in terms of the type of event (manual, event, chained or autotriggered) and the event number. For a description of the bit fields in the queue event entry registers, see [Section 4.2.4.1](#).

2.11.1 DMA/QDMA Channel to Event Queue Mapping

Each of the 64 DMA channels and eight QDMA channels are independently programmed to map to a specific queue, using the DMA queue number register (DMAQNUM) and the QDMA queue number register (QDMANUM). The mapping of DMA/QDMA channels is critical to achieving the desired performance level for the EDMA and most importantly in meeting real-time deadlines (see [Section 2.11.4](#)).

Note: If an event is ready to be queued and both the event queue and the EDMA3 transfer controller associated to the event queue are empty, then the event bypasses the event queue, and goes to the PaRAM processing logic and eventually to the transfer request submission logic for submission to the EDMA3TC. In this case, the event is not logged in the event queue status registers.

2.11.2 Queue RAM Debug Visibility

Each of the 16 entries in the event queue are read using the EDMA3CC memory-mapped register. By reading the event queue, you see the history of the last 16 TRs that have been processed by the EDMA3 on a given queue. This provides user/software visibility and is helpful for debugging real-time issues (typically post-mortem), involving multiple events and event sources.

The event queue entry register (QxEx) 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).

Additionally, the queue status register (QSTAT n) provides a history of the queued events for all queues in the device. The STRTPTR bit in QSTAT n provides the offset to the head entry of an event. The NUMVAL bit in QSTAT n provides the total number of entries residing in the event queue at a given instance of time.

2.11.3 Queue Resource Tracking

The EDMA3CC event queue includes watermarking/threshold logic that allows you to keep track of maximum usage of all event queues. This is useful for debugging real-time deadline violations that may result from head-of-line blocking on a given EDMA3 event queue.

You can program the maximum number of events that can queue up in an event queue by programming the threshold value (between 0 to 15) in the queue watermark threshold A register (QWMTHRA). The maximum queue usage is recorded actively in the watermark (WM) bit of the queue status register (QSTAT n) that keeps getting updated based on a comparison of number of valid entries, which is also visible in the NUMVAL bit in QSTAT n and the maximum number of entries (WM bit in QSTAT n).

If the queue usage is exceeded, this status is visible in the EDMA3CC registers: the QTHRXCDC n bit in the channel controller error register (CCERR) and the THRXCDC bit in QSTAT n , where n stands for the event queue number. Any bits that are set in CCERR also generate an EDMA3CC error interrupt.

2.11.4 Performance Considerations

The main switched central resource (SCR) [see the *TMS320C6457 Fixed-Point Digital Signal Processor* data manual ([SPRS582](#))] arbitrates bus requests from all the masters (CPU, EMAC, RapidIO, UHPI) to the shared slave resources (peripherals and memories).

The priorities of transfer requests (read and write commands) from the EDMA3 transfer controllers with respect to other masters within the system crossbar are programmed using the queue priority register (QUEPRI). QUEPRI programs the priority of the event queues (or indirectly, TC0-TC5, because Queue N transfer requests are submitted to TCM).

Therefore, the priority of unloading queues has a secondary affect compared to the priority of the transfers as they are executed by the EDMA3TC (dictated by the priority set using QUEPRI).

2.12 EDMA3 Transfer Controller (EDMA3TC)

The EDMA3 channel controller is the user-interface of the EDMA3 and the EDMA3 transfer controller (EDMA3TC) is the data movement engine of the EDMA3. The EDMA3CC submits transfer requests (TR) to the EDMA3TC and the EDMA3TC performs the data transfers dictated by the TR; thus, the EDMA3TC is a slave to the EDMA3CC.

2.12.1 Architecture Details

2.12.1.1 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 2.12.5](#).

The EDMA3TC attempts to issue the largest possible command size as limited by the DBS value or the ACNT/BCNT value of the TR. EDMA3TC obeys the following rules:

- The read/write controllers always issue commands less than or equal to the DBS tie-off value
- The first command of a 1D transfer command always aligns to the DBS tie-off value for subsequent commands.

[Table 2-15](#) lists the TR segmentation rules that are followed by the EDMA3TC. In summary, if the ACNT value is larger than the DBS value, then the EDMA3TC breaks the ACNT array into DBS-sized commands to the source/destination addresses. Each BCNT number of arrays are then serviced in succession.

For BCNT arrays of ACNT bytes (that is, a 2D transfer), if the ACNT value is smaller than the DBS value, then the TR may be optimized into a 1D-transfer in order to maximize efficiency. The optimization takes place if the EDMA3TC recognizes that the 2D-transfer is organized as a single dimension (ACNT == BIDX) and the ACNT value is a power of 2.

[Table 2-15](#) lists conditions in which the optimizations are performed. [Example 2-4](#) shows command fragmentation for DBS of 64 bytes.

Table 2-15. 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 |

Example 2-4. Command Fragmentation (DBS = 64)

1. ACNT = 8, BCNT = 8, SRCBIDX = 8, DSTBIDX = 10, SRCADDR = 64, DSTADDR = 191
 Read Controller: This is optimized from a 2D-transfer to a 1D-transfer such that the read side is equivalent to ACNT = 64, BCNT = 1.
 Cmd0 = 64 byte
 Write Controller: Because DSTBIDX != ACNT, it is not optimized.
 Cmd0 = 8 byte
 Cmd1 = 8 byte
 Cmd2 = 8 byte
 Cmd3 = 8 byte
 Cmd4 = 8 byte
 Cmd5 = 8 byte
 Cmd6 = 8 byte
 Cmd7 = 8 byte
2. ACNT=128, BCNT = 1, SRCADDR = 63, DSTADDR = 513
 Read Controller: Read address is not aligned.
 Cmd0 = 1 byte, (now the SRCADDR 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 DSTADDR is aligned to 64 for the next command)
 Cmd1 = 64 bytes
 Cmd2 = 1 byte

2.12.1.2 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.

2.12.1.3 Performance Tuning

By default, reads are as issued as fast as possible. In some cases, the reads issued by the EDMA3TC could fill the available command buffering for a slave, delaying other (potentially higher priority) masters from successfully submitting commands to that slave. The rate at which read commands are issued by the EDMA3TC is controlled by the RDRATE register. The RDRATE register defines the number of cycles that the EDMA3TC read controller waits before issuing subsequent commands for a given TR, thus minimizing the chance of the EDMA3TC consuming all available slave resources. The RDRATE value should be set to a relatively small value 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.

2.12.2 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 EDMA3TC and used by the EDMA3TC while issuing read and write commands. The read or write commands have the same privilege identification, and privilege level as that programmed in the EDMA3 transfer in the channel controller.

2.12.3 Error Generation

Errors are generated if enabled under three conditions:

- EDMA3TC 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 FIFO mode TR violating the FIFO 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 pulsed.

2.12.4 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 EDMA3TC status register (TCSTAT) has dedicated bit fields to indicate the ongoing activity within different parts of the transfer controller:

- The SRCACTV bit indicates whether the source active set is active.
- The DSTACTV bit indicates the number of TRs resident in the destination register active set at a given instance.
- The PROGBUSY bit indicates whether a valid TR is present in the DMA program set.

If the TRs are in progression, caution must be used and you must realize that there is a chance that the values read from the EDMA3TC status registers are inconsistent since the EDMA3TC may change the values of these registers due to ongoing activities.

It is suggested that you ensure no additional submission of TRs to the EDMA3TC in order to facilitate ease of debug.

2.12.4.1 Destination FIFO Register Pointer

The destination FIFO register pointer is implemented as a circular buffer with the start pointer being DFSTRTPTR and a buffer depth of usually 2 or 4. The EDMA3TC maintains two important status details in TCSTAT that may be used during advanced debugging, if necessary. The DFSTRTPTR bit is a start pointer, that is, the index to the head of the destination FIFO register. The DSTACTV bit is a counter for the number of valid (occupied) entries. These registers may be used to get a brief history of transfers.

Examples of some register field values and their interpretation:

- DFSTRTPTR = 0 and DSTACTV = 0 implies that no TRs are stored in the destination FIFO register.
- DFSTRTPTR = 1 and DSTACTV = 2h implies that two TRs are present. The first pending TR is read from the destination FIFO register entry 1 and the second pending TR is read from the destination FIFO register entry 2.
- DFSTRTPTR = 3h and DSTACTV = 2h implies that two TRs are present. The first pending TR is read from the destination FIFO register entry 3 and the second pending TR is read from the destination FIFO register entry 0.

2.12.5 EDMA3TC Configuration

Table 2-16 provides the configuration of the individual EDMA3 transfer controllers present on the device.

Table 2-16. EDMA3 Transfer Controller Configurations

| Name | TC0 | TC1 | TC2 | TC3 | TC4 | TC5 |
|-------------|-----------|-----------|-----------|-----------|-----------|-----------|
| FIFOSIZE | 256 bytes |
| BUSWIDTH | 16 bytes |
| DSTREGDEPTH | 4 entries |
| DBS | 64 bytes |

2.13 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 EDMA3CC activity:

1. Event is asserted from an external source (peripheral or external interrupt). This also is similar for a manually-triggered, chained-triggered, or QDMA-triggered event. The event is latched into the *ER.En/ERH.En* (or *CER.En/CERH.En*, *ESR.En/ESRH.En*, *QER.En*) bit.
2. Once an event is prioritized and queued into the appropriate event queue, the *SER.En/SERH.En* (or *QSER.En*) bit is set to inform the event prioritization/processing logic to disregard this event since it is already in the queue. Alternatively, if the transfer controller and the event queue are empty, then the event bypasses the queue.
3. The EDMA3CC processing and the submission logic evaluates the appropriate PaRAM set and determines whether it is a non-null and non-dummy transfer request (TR).
4. The EDMA3CC clears the *ER.En/ERH.En* (or *CER.En/CERH.En*, *ESR.En/ESRH.En*, *QER.En*) bit and the *SER.En/SERH.En* bit as soon as it determines the TR is non-null. In the case of a null set, the *SER.En/SERH.En* bit remains set. It submits the non-null/non-dummy TR to the associated transfer controller. If the TR was programmed for early completion, the EDMA3CC immediately sets the interrupt pending register (*IPR.I[TCC]/IPRH.I[TCC]-32*).
5. If the TR was programmed for normal completion, the EDMA3CC sets the interrupt pending register (*IPR.I[TCC]/IPRH.I[TCC]*) when the EDMA3TC informs the EDMA3CC about completion of the transfer (returns transfer completion codes).
6. The EDMA3CC programs the associated EDMA3TCn'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 EDMA3TCn.
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 EDMA3TCn then signals completion status to the EDMA3CC.

2.14 EDMA3 Prioritization

The EDMA3 controller has many implementation rules to deal with concurrent events/channels, transfers, etc. The following subsections detail various arbitration details whenever there might be occurrence of concurrent activity. Figure 2-17 shows the different places EDMA3 priorities come into play.

2.14.1 Channel Priority

The DMA event registers (ER and ERH) capture up to 64 events; likewise, the QDMA event register (QER) captures QDMA events for all QDMA channels; therefore, it is possible for events to occur simultaneously on the DMA/QDMA event inputs. For events arriving simultaneously, the event associated with the lowest channel number is prioritized for submission to the event queues (for DMA events, channel 0 has the highest priority and channel 63 has the lowest priority; similarly, for QDMA events, channel 0 has the highest priority and channel 7 has the lowest priority). This mechanism only sorts simultaneous events for submission to the event queues.

If a DMA and QDMA event occurs simultaneously, the DMA event always has prioritization against the QDMA event for submission to the event queues.

2.14.2 Trigger Source Priority

If a DMA channel is associated with more than one trigger source (event trigger, manual trigger, and chain trigger), and if multiple events are set simultaneously for the same channel ($ER.En = 1$, $ESR.En = 1$, $CER.En = 1$), then the EDMA3CC always services these events in the following priority order: event trigger (via ER) is higher priority than chain trigger (via CER) and chain trigger is higher priority than manual trigger (via ESR).

This implies that if for channel 0, both $ER.E0 = 1$ and $CER.E0 = 1$ at the same time, then the $ER.E0$ event is always queued before the $CER.E0$ event.

2.14.3 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 DMAQNUM and QDMAQNUM). For submission of a TR to the transfer request, events need to be dequeued from the event queues. Queue 0 has the highest dequeue priority and queue 3 the lowest.

2.14.4 Transfer Controller Priority

Every transfer controller has a programmed system priority (programmed via QUEPRI) that is implemented when multiple masters in the system are vying for the same endpoint. The priority of the associated transfer request (TR) is further mitigated by the system priority setting of the transfer controller. This priority comes into play at the SCR when several masters are submitting requests to the main SCR.

The default priority for all TCs is the same (0 or highest priority to other masters). If an application requires the EDMA3 to service both real-time (urgent) and nonreal-time transfers, it is recommended that this priority be changed.

2.15 Reset Considerations

A hardware reset resets the EDMA3 (EDMA3CC and EDMA3TC) and the EDMA3 configuration registers. The PaRAM memory contents are undefined after device reset and you should not rely on parameters to be reset to a known state. The PaRAM entry must be initialized to a desired value before it is used.

2.16 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 EDMA3 channel controller and transfer controller operations continue. Events continue to be latched and processed and transfer requests continue to be submitted and serviced.

Since EDMA3 is involved in servicing multiple master and slave peripherals, it is not feasible to have an independent behavior of the EDMA3 for emulation halts. EDMA3 functionality would be coupled with the peripherals it is servicing, which might have different behavior during emulation halts. For example, if a McBSP is halted during an emulation access (FREE = 0 and SOFT = 0 or 1 in McBSP registers), the McBSP stops generating the McBSP receive or transmit events (REVT or XEVT) to the EDMA. From the point of view of the McBSP, the EDMA3 is suspended, but other peripherals (for example, a timer) still assert events and be serviced by the EDMA.

EDMA3 Transfer Examples

The EDMA3 channel controller performs a variety of transfers depending on the parameter configuration. The following sections provides a description and PaRAM configuration for some typical use case scenarios.

| Topic | Page |
|--|-------------|
| 3.1 Block Move Example | 70 |
| 3.2 Subframe Extraction Example | 72 |
| 3.3 Data Sorting Example | 74 |
| 3.4 Peripheral Servicing Example..... | 76 |

Figure 3-2. Block Move Example PaRAM Configuration

(a) EDMA Parameters

| Parameter Contents | | Parameter | |
|--------------------|-------|-----------------------------------|--------------------------------|
| 0010 0008h | | Channel Options Parameter (OPT) | |
| 8000 0000h | | Channel Source Address (SRC) | |
| 0001h | 0100h | Count for 2nd Dimension (BCNT) | Count for 1st Dimension (ACNT) |
| 1080 0000h | | Channel Destination Address (DST) | |
| 0000h | 0000h | Destination BCNT Index (DSTBIDX) | Source BCNT Index (SRCBIDX) |
| 0000h | FFFFh | BCNT Reload (BCNTRLD) | Link Address (LINK) |
| 0000h | 0000h | Destination CCNT Index (DSTCIDX) | Source CCNT Index (SRCCIDX) |
| 0000h | 0001h | Reserved | Count for 3rd Dimension (CCNT) |

(b) Channel Options Parameter (OPT) Content

| | | | | | | | | | | | | |
|------|----------|--------|----------|--------|----------|---------|----------|-----|----|----|----|----|
| 31 | 30 | 28 | 27 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| 0 | 000 | 0000 | 0 | 0 | 0 | 1 | 00 | 00 | | | | |
| PRIV | Reserved | PRIVID | ITCCHEN | TCCHEN | ITCINTEN | TCINTEN | Reserved | TCC | | | | |
| 15 | 12 | 11 | 10 | 8 | 7 | 4 | 3 | 2 | 1 | 0 | | |
| 0000 | 0 | 000 | 0000 | 1 | 0 | 0 | 0 | | | | | |
| TCC | TCCMOD | FWID | Reserved | STATIC | SYNCDIM | DAM | SAM | | | | | |

3.2 Subframe Extraction Example

The EDMA3 can efficiently extract a small frame of data from a larger frame of data. By performing a 2D-to-1D transfer, the EDMA3 retrieves a portion of data for the CPU to process. In this example, a 640×480 -pixel frame of video data is stored in external memory. Each pixel is represented by a 16-bit halfword. The CPU extracts a 16×12 -pixel subframe of the image for processing. To facilitate more efficient processing time by the CPU, the EDMA3 places the subframe in internal L2 SRAM of C64X+ core0. Figure 3-3 shows the transfer of a subframe from external memory to L2. Figure 3-4 shows the parameters for this transfer.

The same PaPARAM entry options are used for QDMA channels, as well as DMA channels. The STATIC bit in OPT is set to prevent linking. For successive transfers, only changed parameters need to be programmed before triggering the channel.

Figure 3-3. Subframe Extraction Example

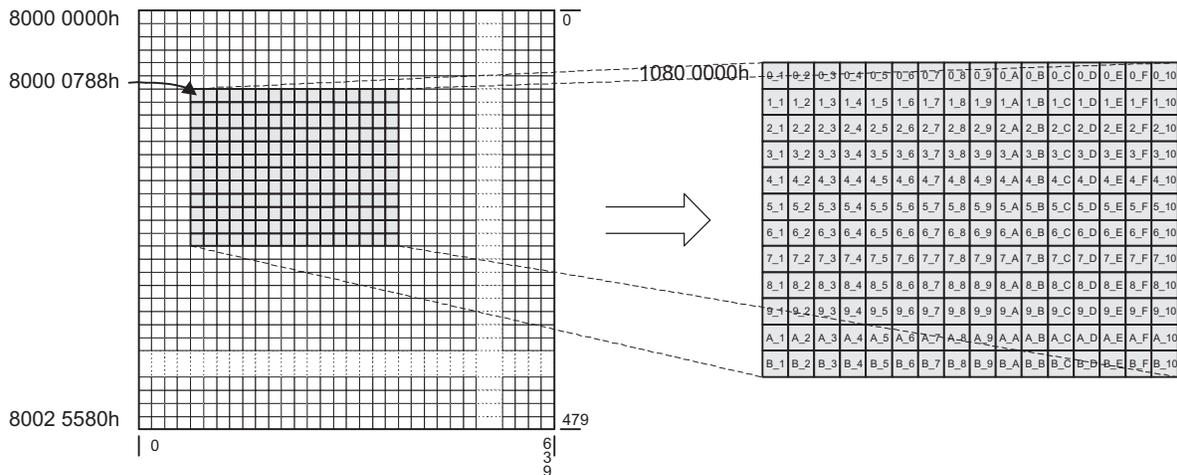


Figure 3-4. Subframe Extraction Example PaRAM Configuration

(a) EDMA Parameters

| Parameter Contents | | Parameter | |
|--------------------|-------|-----------------------------------|--------------------------------|
| 0010 000Ch | | Channel Options Parameter (OPT) | |
| 8000 0788h | | Channel Source Address (SRC) | |
| 000Ch | 0020h | Count for 2nd Dimension (BCNT) | Count for 1st Dimension (ACNT) |
| 1080 0000h | | Channel Destination Address (DST) | |
| 0020h | 0500h | Destination BCNT Index (DSTBIDX) | Source BCNT Index (SRCBIDX) |
| 0000h | FFFFh | BCNT Reload (BCNTRLD) | Link Address (LINK) |
| 0000h | 0000h | Destination CCNT Index (DSTCIDX) | Source CCNT Index (SRCCIDX) |
| 0000h | 0001h | Reserved | Count for 3rd Dimension (CCNT) |

(b) Channel Options Parameter (OPT) Content

| | | | | | | | | | | | | |
|------|----------|--------|----------|--------|----------|---------|----------|-----|----|----|----|----|
| 31 | 30 | 28 | 27 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| 0 | 000 | 0000 | 0 | 0 | 0 | 1 | 00 | 00 | | | | |
| PRIV | Reserved | PRIVID | ITCCHEN | TCCHEN | ITCINTEN | TCINTEN | Reserved | TCC | | | | |
| 15 | 12 | 11 | 10 | 8 | 7 | 4 | 3 | 2 | 1 | 0 | | |
| 0000 | 0 | 000 | 0000 | 1 | 1 | 0 | 0 | | | | | |
| TCC | TCCMOD | FWID | Reserved | STATIC | SYNCDIM | DAM | SAM | | | | | |

3.3 Data Sorting Example

Many applications require the use of multiple data arrays; it is often desirable to have the arrays arranged such that the first elements of each array are adjacent, the second elements are adjacent, and so on. Often this is not how the data is presented to the device. Either data is transferred via a peripheral with the data arrays arriving one after the other or the arrays are located in memory with each array occupying a portion of contiguous memory spaces. For these instances, the EDMA3 can reorganize the data into the desired format. [Figure 3-5](#) shows the data sorting.

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.
- SRCBIDX - Program this to be the size of the element or ACNT.
- DSTBIDX - CCNT × ACNT
- SRCCDX - ACNT × BCNT
- DSTCIDX - ACNT

The synchronization type needs to be AB-synchronized and the STATIC bit is 0 to allow updates to the parameter set. It is advised to use normal EDMA3 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 3-6](#) shows the parameter set programming for this transfer, assuming channel 0 and an element size of 4 bytes.

Figure 3-5. Data Sorting Example

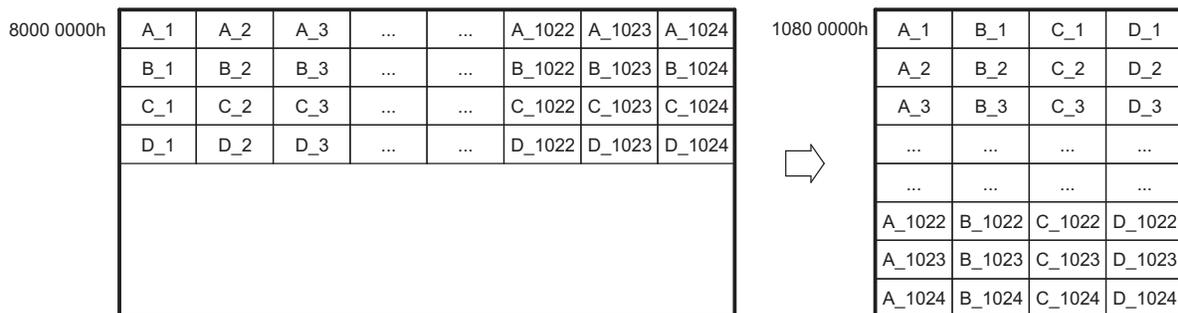


Figure 3-6. Data Sorting Example PaRAM Configuration

(a) EDMA Parameters

| Parameter Contents | | Parameter | |
|--------------------|-------|-----------------------------------|--------------------------------|
| 0090 0004h | | Channel Options Parameter (OPT) | |
| 8000 0000h | | Channel Source Address (SRC) | |
| 0400h | 0004h | Count for 2nd Dimension (BCNT) | Count for 1st Dimension (ACNT) |
| 1080 0000h | | 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

| 31 | 30 | 28 | 27 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|------|----------|--------|------|----------|--------|----------|---------|----------|-----|-----|----|----|
| 0 | 000 | 0000 | | 1 | 0 | 0 | 1 | 00 | | 00 | | |
| PRIV | Reserved | PRIVID | | ITCCHEN | TCCHEN | ITCINTEN | TCINTEN | Reserved | | TCC | | |
| 15 | 12 | 11 | 10 | 8 | 7 | 4 | | 3 | 2 | 1 | 0 | |
| 0000 | | 0 | 000 | 0000 | | | 0 | 1 | 0 | 0 | | |
| TCC | | TCCMOD | FWID | Reserved | | | STATIC | SYNCDIM | DAM | SAM | | |

Figure 3-8. Servicing Incoming McBSP Data Example PaRAM Configuration

(a) EDMA Parameters

| Parameter Contents | | Parameter | |
|--------------------|-------|-----------------------------------|--------------------------------|
| 0010 0000h | | Channel Options Parameter (OPT) | |
| 3000 0000h | | Channel Source Address (SRC) | |
| 0100h | 0001h | Count for 2nd Dimension (BCNT) | Count for 1st Dimension (ACNT) |
| 1080 0000h | | Channel Destination Address (DST) | |
| 0001h | 0000h | Destination BCNT Index (DSTBIDX) | Source BCNT Index (SRCBIDX) |
| 0000h | FFFFh | BCNT Reload (BCNTRLD) | Link Address (LINK) |
| 0000h | 0000h | Destination CCNT Index (DSTCIDX) | Source CCNT Index (SRCCIDX) |
| 0000h | 0004h | Reserved | Count for 3rd Dimension (CCNT) |

(b) Channel Options Parameter (OPT) Content

| | | | | | | | | | | | | |
|------|----------|--------|------|----------|--------|----------|---------|----------|-----|-----|----|----|
| 31 | 30 | 28 | 27 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| 0 | 000 | 0000 | | 0 | 0 | 0 | 1 | 00 | | 00 | | |
| PRIV | Reserved | PRIVID | | ITCCHEN | TCCHEN | ITCINTEN | TCINTEN | Reserved | | TCC | | |
| | | | | | | | | | | | | |
| 15 | 12 | 11 | 10 | 8 | 7 | 4 | | 3 | 2 | 1 | 0 | |
| 0000 | | 0 | 000 | 0000 | | | 0 | 0 | 0 | 0 | | |
| TCC | | TCCMOD | FWID | Reserved | | | STATIC | SYNCDIM | DAM | SAM | | |

3.4.2 Bursting Peripherals

Higher bandwidth applications require that multiple data elements be presented to the DSP 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 DSP.

In this example, a video frame 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. Figure 3-9 shows this example.

To transfer data from an external peripheral to an external buffer one array at a time based on EVT_n , channel n must be configured. Due to the nature of the data (a video frame made up of arrays of pixels) the destination is essentially a 2D entity. Figure 3-10 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 SRCBIDX is 0 (no modification to the source address). The destination is made up of arrays of contiguous, linear elements; therefore, the DSTBIDX is set to pixel size, 2 bytes. ANCT is equal to the pixel size, 2 bytes. BCNT is set to the number of pixels in an array, 640. CCNT is equal to the total number of arrays in the block, 480. SRCCIDX is 0 because the source address undergoes no increment. The DSTCIDX is equal to the difference between the starting addresses of each array. Because a pixel is 16 bits (2 bytes), DSTCIDX is equal to 640×2 .

Figure 3-9. Servicing Peripheral Burst Example

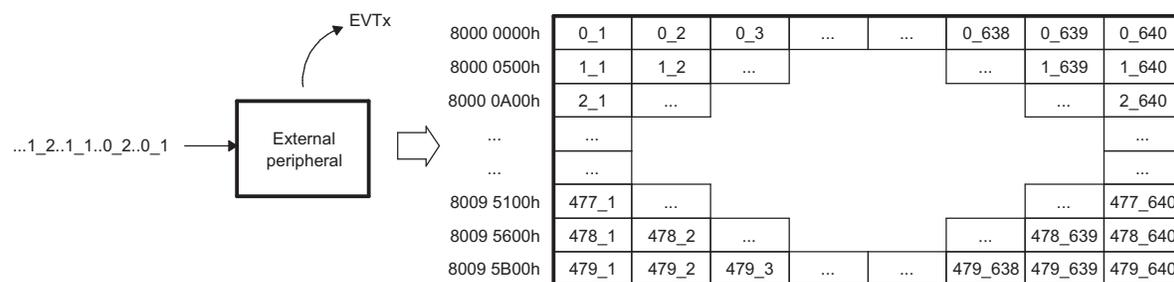


Figure 3-10. Servicing Peripheral Burst Example PaRAM Configuration

(a) EDMA Parameters

| Parameter Contents | | Parameter | |
|------------------------|-------|-----------------------------------|--------------------------------|
| 0010 0004h | | Channel Options Parameter (OPT) | |
| Channel Source Address | | Channel Source Address (SRC) | |
| 0280h | 0002h | Count for 2nd Dimension (BCNT) | Count for 1st Dimension (ACNT) |
| 8000 0000h | | Channel Destination Address (DST) | |
| 0002h | 0000h | Destination BCNT Index (DSTBIDX) | Source BCNT Index (SRCBIDX) |
| 0000h | FFFFh | BCNT Reload (BCNTRLD) | Link Address (LINK) |
| 0500h | 0000h | Destination CCNT Index (DSTCIDX) | Source CCNT Index (SRCCIDX) |
| 0000h | 01E0h | Reserved | Count for 3rd Dimension (CCNT) |

(b) Channel Options Parameter (OPT) Content

| | | | | | | | | | | | | |
|------|----------|--------|----------|----------|----------|----------|----------|---------|-----|-----|----|----|
| 31 | 30 | 28 | 27 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| 0 | 000 | 0000 | 0 | 0 | 0 | 1 | 00 | 00 | | | | |
| PRIV | Reserved | PRIVID | ITCCHEN | TCCHEN | ITCINTEN | TCINTEN | Reserved | TCC | | | | |
| 15 | 12 | 11 | 10 | 8 | 7 | 4 | 3 | 2 | 1 | 0 | | |
| 0000 | 0 | 000 | 0000 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | | |
| TCC | TCCMOD | FWID | Reserved | Reserved | Reserved | Reserved | STATIC | SYNCDIM | DAM | SAM | | |

3.4.3 Continuous Operation

Configuring an EDMA3 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 DSP. In this case, it is necessary to implement some form of linking such that the EDMA3 channels continuously reload the necessary parameter sets. In this example, McBSP0 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 L2 memory and from L2 memory to the serial port, as shown [Figure 3-11](#).

The McBSP generates REVT0 for every element received and generates XEVT0 for every element transmitted. To service the data streams, EDMA3 channels 12 and 13 must be set up for 1D-to-1D transfers with A-synchronization.

[Figure 3-12](#) shows the parameter entries for the channel for these transfers. To service the McBSP continuously throughout DSP operation, the channels must be linked to a duplicate PaRAM set in the PaRAM. After all frames have been transferred, the EDMA3 channels reload and continue. [Figure 3-13](#) shows the reload parameters for the channel.

3.4.3.1 Receive Channel

EDMA3 channel 13 services the incoming data stream of McBSP0. The source address is set to that of the data receiver register (DRR), and the destination address is set to the first element of the data block. 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 13 parameter set, the parameters located at the link address are loaded into the channel 13 parameter set and operation continues. This function continues throughout device operation until halted by the CPU.

3.4.3.2 Transmit Channel

EDMA3 channel 12 services the outgoing data stream of McBSP0. 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 EDMA3 channel, with duplicate PaRAM set entries at PaRAM set 65.

Figure 3-11. Servicing Continuous McBSP Data Example

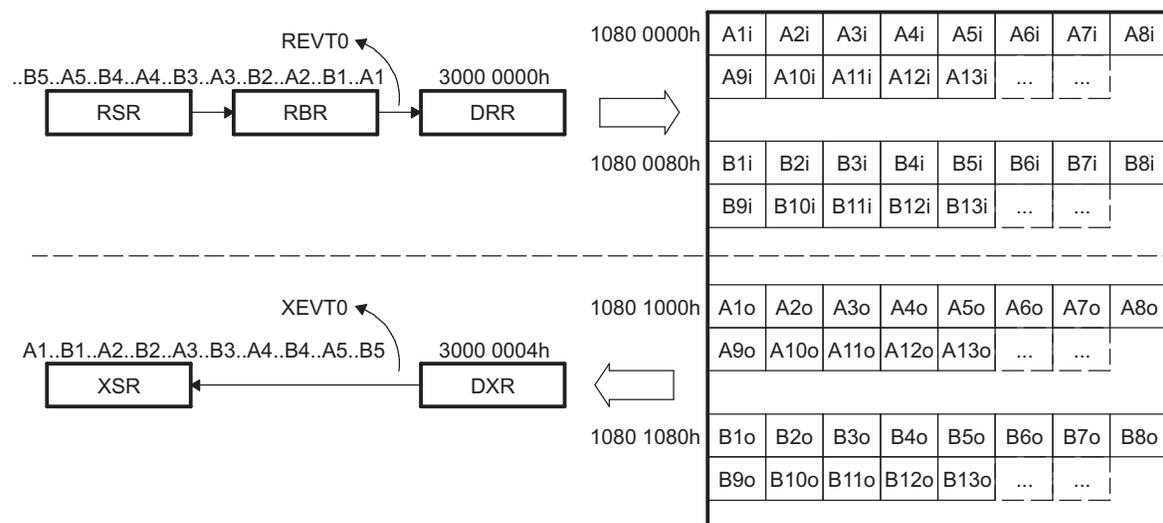


Figure 3-12. Servicing Continuous McBSP Data Example PaRAM Configuration

(a) EDMA Parameters for Receive Channel (PaRAM Set 13) being Linked to PaRAM Set 64

| Parameter Contents | | Parameter | |
|--------------------|-------|-----------------------------------|--------------------------------|
| 0010 0000h | | Channel Options Parameter (OPT) | |
| 3000 0000h | | Channel Source Address (SRC) | |
| 0080h | 0001h | Count for 2nd Dimension (BCNT) | Count for 1st Dimension (ACNT) |
| 1080 0000h | | Channel Destination Address (DST) | |
| 0001h | 0000h | Destination BCNT Index (DSTBIDX) | Source BCNT Index (SRCBIDX) |
| 0080h | 4800h | BCNT Reload (BCNTRLD) | Link Address (LINK) |
| 0000h | 0000h | Destination CCNT Index (DSTCIDX) | Source CCNT Index (SRCCIDX) |
| 0000h | FFFFh | Reserved | Count for 3rd Dimension (CCNT) |

(b) Channel Options Parameter (OPT) Content for Receive Channel (PaRAM Set 13)

| | | | | | | | | | | | | |
|------|----------|--------|----------|--------|----------|---------|----------|-----|----|----|----|----|
| 31 | 30 | 28 | 27 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| 0 | 000 | 0000 | 0 | 0 | 0 | 1 | 00 | 00 | | | | |
| PRIV | Reserved | PRIVID | ITCCHEN | TCCHEN | ITCINTEN | TCINTEN | Reserved | TCC | | | | |
| 15 | 12 | 11 | 10 | 8 | 7 | 4 | 3 | 2 | 1 | 0 | | |
| 0000 | 0 | 000 | 0000 | 0 | 0 | 0 | 0 | | | | | |
| TCC | TCCMOD | FWID | Reserved | STATIC | SYNCDIM | DAM | SAM | | | | | |

) EDMA Parameters for Transmit Channel (PaRAM Set 12) being Linked to PaRAM Set 65

| Parameter Contents | | Parameter | |
|--------------------|-------|-----------------------------------|--------------------------------|
| 0010 1000h | | Channel Options Parameter (OPT) | |
| 1080 1000h | | Channel Source Address (SRC) | |
| 0080h | 0001h | Count for 2nd Dimension (BCNT) | Count for 1st Dimension (ACNT) |
| 3000 0004h | | Channel Destination Address (DST) | |
| 0000h | 0001h | Destination BCNT Index (DSTBIDX) | Source BCNT Index (SRCBIDX) |
| 0080h | 4820h | BCNT Reload (BCNTRLD) | Link Address (LINK) |
| 0000h | 0000h | Destination CCNT Index (DSTCIDX) | Source CCNT Index (SRCCIDX) |
| 0000h | FFFFh | Reserved | Count for 3rd Dimension (CCNT) |

(d) Channel Options Parameter (OPT) Content for Transmit Channel (PaRAM Set 12)

| | | | | | | | | | | | | |
|------|----------|--------|----------|--------|----------|---------|----------|-----|----|----|----|----|
| 31 | 30 | 28 | 27 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| 0 | 000 | 0000 | 0 | 0 | 0 | 1 | 00 | 00 | | | | |
| PRIV | Reserved | PRIVID | ITCCHEN | TCCHEN | ITCINTEN | TCINTEN | Reserved | TCC | | | | |
| 15 | 12 | 11 | 10 | 8 | 7 | 4 | 3 | 2 | 1 | 0 | | |
| 0001 | 0 | 000 | 0000 | 0 | 0 | 0 | 0 | | | | | |
| TCC | TCCMOD | FWID | Reserved | STATIC | SYNCDIM | DAM | SAM | | | | | |

Figure 3-13. Servicing Continuous McBSP Data Example Reload PaRAM Configuration

(a) EDMA Reload Parameters (PaRAM Set 64) for Receive Channel

| Parameter Contents | | Parameter | |
|--------------------|-------|-----------------------------------|--------------------------------|
| 0010 0000h | | Channel Options Parameter (OPT) | |
| 3000 0000h | | Channel Source Address (SRC) | |
| 0080h | 0001h | Count for 2nd Dimension (BCNT) | Count for 1st Dimension (ACNT) |
| 1080 0000h | | Channel Destination Address (DST) | |
| 0001h | 0000h | Destination BCNT Index (DSTBIDX) | Source BCNT Index (SRCBIDX) |
| 0080h | 4800h | BCNT Reload (BCNTRLD) | Link Address (LINK) |
| 0000h | 0000h | Destination CCNT Index (DSTCIDX) | Source CCNT Index (SRCCIDX) |
| 0000h | FFFFh | Reserved | Count for 3rd Dimension (CCNT) |

(b) Channel Options Parameter (OPT) Content for Receive Channel (PaRAM Set 64)

| | | | | | | | | | | | | |
|------|----------|--------|----------|--------|----------|---------|----------|-----|----|----|----|----|
| 31 | 30 | 28 | 27 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| 0 | 000 | 0000 | 0 | 0 | 0 | 1 | 00 | 00 | | | | |
| PRIV | Reserved | PRIVID | ITCCHEN | TCCHEN | ITCINTEN | TCINTEN | Reserved | TCC | | | | |
| 15 | 12 | 11 | 10 | 8 | 7 | 4 | 3 | 2 | 1 | 0 | | |
| 0000 | 0 | 000 | 0000 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | |
| TCC | TCCMOD | FWID | Reserved | STATIC | SYNCDIM | DAM | SAM | | | | | |

(c) EDMA Reload Parameters (PaRAM Set 65) for Transmit Channel

| Parameter Contents | | Parameter | |
|--------------------|-------|-----------------------------------|--------------------------------|
| 0010 1000h | | Channel Options Parameter (OPT) | |
| 1080 1000h | | Channel Source Address (SRC) | |
| 0080h | 0001h | Count for 2nd Dimension (BCNT) | Count for 1st Dimension (ACNT) |
| 3000 0004h | | Channel Destination Address (DST) | |
| 0000h | 0001h | Destination BCNT Index (DSTBIDX) | Source BCNT Index (SRCBIDX) |
| 0080h | 4820h | BCNT Reload (BCNTRLD) | Link Address (LINK) |
| 0000h | 0000h | Destination CCNT Index (DSTCIDX) | Source CCNT Index (SRCCIDX) |
| 0000h | FFFFh | Reserved | Count for 3rd Dimension (CCNT) |

(d) Channel Options Parameter (OPT) Content for Transmit Channel (PaRAM Set 65)

| | | | | | | | | | | | | |
|------|----------|--------|----------|--------|----------|---------|----------|-----|----|----|----|----|
| 31 | 30 | 28 | 27 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| 0 | 000 | 0000 | 0 | 0 | 0 | 1 | 00 | 00 | | | | |
| PRIV | Reserved | PRIVID | ITCCHEN | TCCHEN | ITCINTEN | TCINTEN | Reserved | TCC | | | | |
| 15 | 12 | 11 | 10 | 8 | 7 | 4 | 3 | 2 | 1 | 0 | | |
| 0001 | 0 | 000 | 0000 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | |
| TCC | TCCMOD | FWID | Reserved | STATIC | SYNCDIM | DAM | SAM | | | | | |

3.4.4 Ping-Pong Buffering

Although the previous configuration allows the EDMA3 to service a peripheral continuously, it presents a number of restrictions to the CPU. Because the input and output buffers are continuously being filled/emptied, the CPU must match the pace of the EDMA3 very closely to process the data. The EDMA3 receive data must always be placed in memory before the CPU accesses it, and the CPU must provide the output data before the EDMA3 transfers it. Though not impossible, this is an unnecessary challenge. It is particularly difficult in a 2-level cache scheme.

Ping-pong buffering is a simple technique that allows the CPU activity to be distanced from the EDMA3 activity. This means that there are multiple (usually two) sets of data buffers for all incoming and outgoing data streams. While the EDMA3 transfers the data into and out of the ping buffers, the CPU manipulates the data in the pong buffers. When both CPU and EDMA3 activity completes, they switch. The EDMA3 then writes over the old input data and transfers the new output data. [Figure 3-14](#) shows the ping-pong scheme for this example.

To change the continuous operation example, such that a ping-pong buffering scheme is used, the EDMA3 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 3-15](#) shows the EDMA3 channel configuration required.

Each channel has two parameter sets, ping and pong. The EDMA3 channel is initially loaded with the ping parameters ([Figure 3-15](#)). The link address for the ping set is set to the PaRAM offset of the pong parameter set ([Figure 3-16](#)). The link address for the pong set is set to the PaRAM offset of the ping parameter set ([Figure 3-17](#)). 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.

3.4.4.1 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 EDMA3 to complete before switching to the alternate (pong) buffer. In this example, both channels provide their channel numbers as their report word and set the TCINTEN bit to generate an interrupt after completion. When channel 13 fills an input buffer, the E13 bit in the interrupt pending register (IPR) is set; when channel 12 empties an output buffer, the E12 bit in IPR is set. The CPU must manually clear these bits. With the channel parameters set, the CPU polls IPR to determine when to switch. The EDMA3 and CPU could alternatively be configured such that the channel completion interrupts the CPU. By doing this, the CPU could service a background task while waiting for the EDMA3 to complete.

Figure 3-14. Ping-Pong Buffering for McBSP Data Example

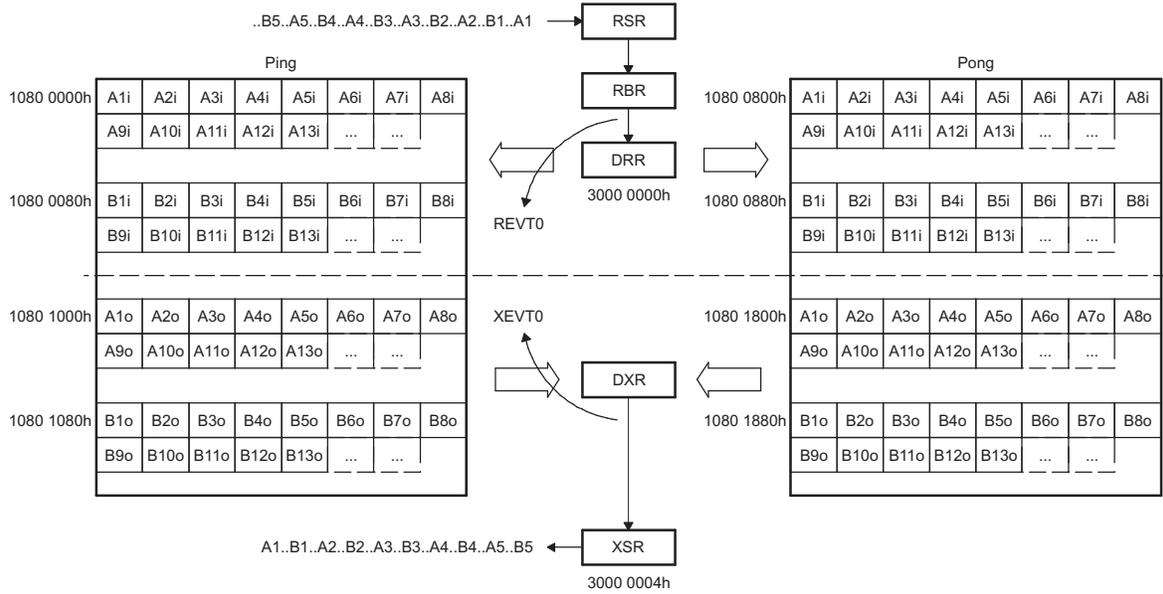


Figure 3-15. Ping-Pong Buffering for McBSP Example PaRAM Configuration

(a) EDMA Parameters for Channel 13 (Using PaRAM Set 13 Linked to Pong Set 64)

| Parameter Contents | | Parameter | |
|--------------------|-------|-----------------------------------|--------------------------------|
| 0010 D000h | | Channel Options Parameter (OPT) | |
| 3000 0000h | | Channel Source Address (SRC) | |
| 0080h | 0001h | Count for 2nd Dimension (BCNT) | Count for 1st Dimension (ACNT) |
| 1080 0000h | | Channel Destination Address (DST) | |
| 0001h | 0000h | Destination BCNT Index (DSTBIDX) | Source BCNT Index (SRCBIDX) |
| 0080h | 4800h | BCNT Reload (BCNTRLD) | Link Address (LINK) |
| 0000h | 0000h | Destination CCNT Index (DSTCIDX) | Source CCNT Index (SRCCIDX) |
| 0000h | 0001h | Reserved | Count for 3rd Dimension (CCNT) |

(b) Channel Options Parameter (OPT) Content for Channel 13

| | | | | | | | | | | | | |
|------|----------|--------|----------|--------|----------|---------|----------|-----|----|----|----|----|
| 31 | 30 | 28 | 27 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| 0 | 000 | 0000 | 0 | 0 | 0 | 1 | 00 | 00 | | | | |
| PRIV | Reserved | PRIVID | ITCCHEN | TCCHEN | ITCINTEN | TCINTEN | Reserved | TCC | | | | |
| 15 | 12 | 11 | 10 | 8 | 7 | 4 | 3 | 2 | 1 | 0 | | |
| 1101 | 0 | 000 | 0000 | 0 | 0 | 0 | 0 | | | | | |
| TCC | TCCMOD | FWID | Reserved | STATIC | SYNCDIM | DAM | SAM | | | | | |

(c) EDMA Parameters for Channel 12 (Using PaRAM Set 12 Linked to Pong Set 65)

| Parameter Contents | | Parameter | |
|--------------------|-------|-----------------------------------|--------------------------------|
| 0010 C000h | | Channel Options Parameter (OPT) | |
| 1080 1000h | | Channel Source Address (SRC) | |
| 0080h | 0001h | Count for 2nd Dimension (BCNT) | Count for 1st Dimension (ACNT) |
| 3000 0004h | | Channel Destination Address (DST) | |
| 0000h | 0001h | Destination BCNT Index (DSTBIDX) | Source BCNT Index (SRCBIDX) |
| 0080h | 4840h | BCNT Reload (BCNTRLD) | Link Address (LINK) |
| 0000h | 0000h | Destination CCNT Index (DSTCIDX) | Source CCNT Index (SRCCIDX) |
| 0000h | 0001h | Reserved | Count for 3rd Dimension (CCNT) |

(d) Channel Options Parameter (OPT) Content for Channel 12

| | | | | | | | | | | | | |
|------|----------|--------|----------|--------|----------|---------|----------|-----|----|----|----|----|
| 31 | 30 | 28 | 27 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| 0 | 000 | 0000 | 0 | 0 | 0 | 1 | 00 | 00 | | | | |
| PRIV | Reserved | PRIVID | ITCCHEN | TCCHEN | ITCINTEN | TCINTEN | Reserved | TCC | | | | |
| 15 | 12 | 11 | 10 | 8 | 7 | 4 | 3 | 2 | 1 | 0 | | |
| 1100 | 0 | 000 | 0000 | 0 | 0 | 0 | 0 | | | | | |
| TCC | TCCMOD | FWID | Reserved | STATIC | SYNCDIM | DAM | SAM | | | | | |

Figure 3-16. Ping-Pong Buffering for McBSP Example Pong PaRAM Configuration

(a) EDMA Pong Parameters for Channel 13 at Set 64 Linked to Set 65

| Parameter Contents | | Parameter | |
|--------------------|-------|-----------------------------------|--------------------------------|
| 0010 D000h | | Channel Options Parameter (OPT) | |
| 3000 0000h | | Channel Source Address (SRC) | |
| 0080h | 0001h | Count for 2nd Dimension (BCNT) | Count for 1st Dimension (ACNT) |
| 1080 0800h | | Channel Destination Address (DST) | |
| 0001h | 0000h | Destination BCNT Index (DSTBIDX) | Source BCNT Index (SRCBIDX) |
| 0080h | 4820h | BCNT Reload (BCNTRLD) | Link Address (LINK) |
| 0000h | 0000h | Destination CCNT Index (DSTCIDX) | Source CCNT Index (SRCCIDX) |
| 0000h | 0001h | Reserved | Count for 3rd Dimension (CCNT) |

(b) EDMA Pong Parameters for Channel 12 at Set 66 Linked to Set 67

| Parameter Contents | | Parameter | |
|--------------------|-------|-----------------------------------|--------------------------------|
| 0010 C000h | | Channel Options Parameter (OPT) | |
| 1080 1800h | | Channel Source Address (SRC) | |
| 0080h | 0001h | Count for 2nd Dimension (BCNT) | Count for 1st Dimension (ACNT) |
| 3000 0004h | | Channel Destination Address (DST) | |
| 0000h | 0001h | Destination BCNT Index (DSTBIDX) | Source BCNT Index (SRCBIDX) |
| 0080h | 4860h | BCNT Reload (BCNTRLD) | Link Address (LINK) |
| 0000h | 0000h | Destination CCNT Index (DSTCIDX) | Source CCNT Index (SRCCIDX) |
| 0000h | 0001h | Reserved | Count for 3rd Dimension (CCNT) |

Figure 3-17. Ping-Pong Buffering for McBSP Example Ping PaRAM Configuration

(a) EDMA Ping Parameters for Channel 13 at Set 65 Linked to Set 64

| Parameter Contents | | Parameter | |
|--------------------|-------|-----------------------------------|--------------------------------|
| 0010 D000h | | Channel Options Parameter (OPT) | |
| 3000 0000h | | Channel Source Address (SRC) | |
| 0080h | 0001h | Count for 2nd Dimension (BCNT) | Count for 1st Dimension (ACNT) |
| 1080 0000h | | Channel Destination Address (DST) | |
| 0001h | 0000h | Destination BCNT Index (DSTBIDX) | Source BCNT Index (SRCBIDX) |
| 0080h | 4800h | BCNT Reload (BCNTRLD) | Link Address (LINK) |
| 0000h | 0000h | Destination CCNT Index (DSTCIDX) | Source CCNT Index (SRCCIDX) |
| 0000h | 0001h | Reserved | Count for 3rd Dimension (CCNT) |

(b) EDMA Ping Parameters for Channel 12 at Set 67 Linked to Set 66

| Parameter Contents | | Parameter | |
|--------------------|-------|-----------------------------------|--------------------------------|
| 0010 C000h | | Channel Options Parameter (OPT) | |
| 1080 1000h | | Channel Source Address (SRC) | |
| 0080h | 0001h | Count for 2nd Dimension (BCNT) | Count for 1st Dimension (ACNT) |
| 3000 0004h | | Channel Destination Address (DST) | |
| 0000h | 0001h | Destination BCNT Index (DSTBIDX) | Source BCNT Index (SRCBIDX) |
| 0080h | 4840h | BCNT Reload (BCNTRLD) | Link Address (LINK) |
| 0000h | 0000h | Destination CCNT Index (DSTCIDX) | Source CCNT Index (SRCCIDX) |
| 0000h | 0001h | Reserved | Count for 3rd Dimension (CCNT) |

3.4.5 Transfer Chaining Examples

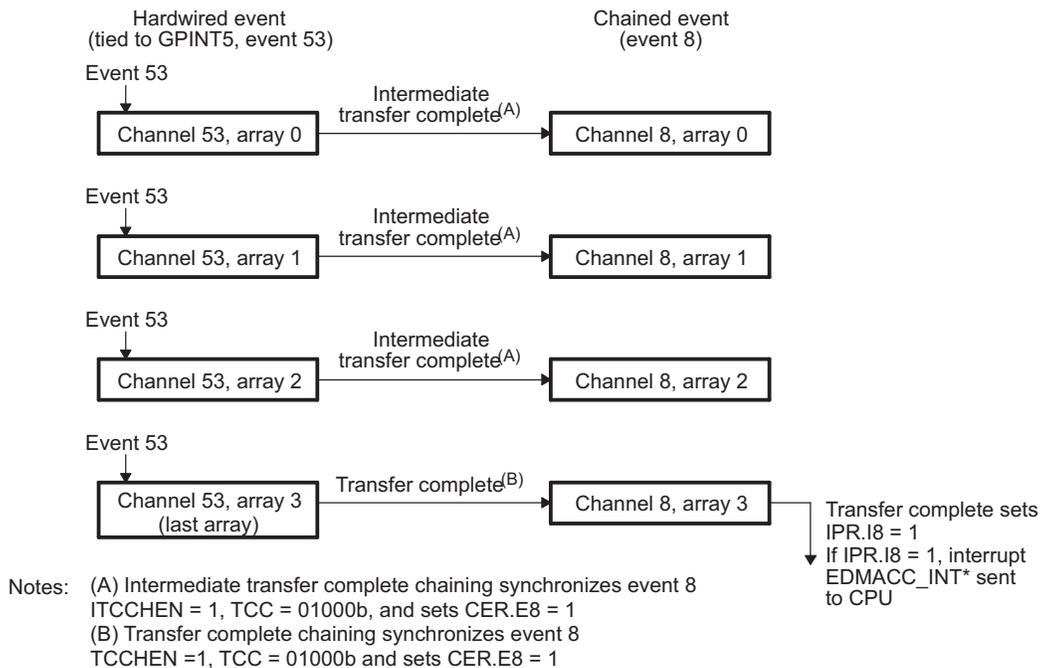
The following examples explain the intermediate transfer complete chaining function.

3.4.5.1 Servicing Input/Output FIFOs with a Single Event

Many systems require the use of a pair of external FIFOs that must be serviced at the same rate. One FIFO buffers data input, and the other buffers data output. The EDMA3 channels that service these FIFOs can be set up for AB-synchronized transfers. While each FIFO is serviced with a different set of parameters, both can be signaled from a single event. For example, an external interrupt pin can be tied to the status flags of one of the FIFOs. When this event arrives, the EDMA3 needs to perform servicing for both the input and output streams. Without the intermediate transfer complete chaining feature this would require two events, and thus two external interrupt pins. The intermediate transfer complete chaining feature allows the use of a single external event (for example, a GPIO event). [Figure 3-18](#) shows the EDMA3 setup and illustration for this example.

A GPIO event (in this case, GPINT5) triggers an array transfer. Upon completion of each intermediate array transfer of channel 53, intermediate transfer complete chaining sets the E8 bit (specified by TCC of 8) in the chained event register (CER) and provides a synchronization event to channel 8. Upon completion of the last array transfer of channel 53, transfer complete chaining—not intermediate transfer complete chaining—sets the E8 bit in CER (specified by TCCMODE:TCC) and provides a synchronization event to channel 8. The completion of channel 8 sets the I8 bit (specified by TCCMODE:TCC) in the interrupt pending register (IPR), which can generate an interrupt to the CPU, if the I8 bit in the interrupt enable register (IER) is set.

Figure 3-18. Intermediate Transfer Completion Chaining Example



Setup

Channel 53 parameters for chaining

- Enable transfer complete chaining:
OPT.TCCHEN = 1
OPT.TCC = 01000b
- Enable intermediate transfer complete chaining:
OPT.ITCCHEN = 1
OPT.TCC = 01000b

Channel 8 parameters for chaining

- Enable transfer complete chaining:
OPT.TCINTEN = 1
OPT.TCC = 01000b
- Disable intermediate transfer complete chaining:
OPT.ITCCHEN = 0

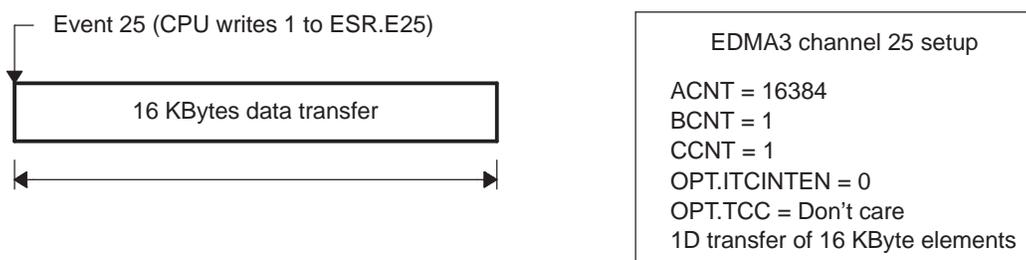
Event enable register (EER)

- Enable channel 53
EER.E53 = 1

3.4.5.2 Breaking Up Large Transfers with Intermediate Chaining

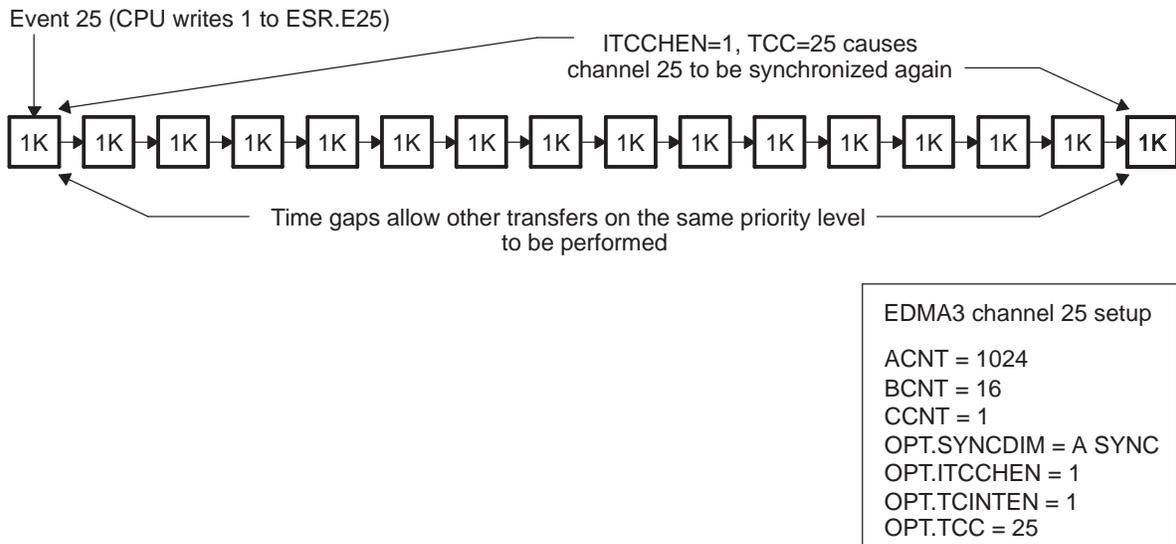
Another feature of intermediate transfer chaining (ITCCHEN) is for breaking up large transfers. A large transfer may lock out other transfers of the same priority level for the duration of the transfer. For example, a large transfer on queue 0 from the internal memory to the external memory using the EMIF may starve other EDMA3 transfers on the same queue. In addition, this large high-priority transfer may prevent the EMIF for a long duration to service other lower priority transfers. When a large transfer is considered to be high priority, it should be split into multiple smaller transfers. Figure 3-19 shows the EDMA3 setup and illustration of an example single large block transfer.

Figure 3-19. Single Large Block Transfer Example



The intermediate transfer chaining enable (ITCCHEN) provides a method to break up a large transfer into smaller transfers. For example, to move a single large block of memory (16K bytes), the EDMA3 performs an A-synchronized transfer. The element count is set to a reasonable value, where reasonable derives from the amount of time it would take to move this smaller amount of data. Assume 1 Kbyte is a reasonable small transfer in this example. The EDMA3 is set up to transfer 16 arrays of 1 Kbyte elements, for a total of 16K byte elements. The TCC field in the channel options parameter (OPT) is set to the same value as the channel number and ITCCHEN are set. In this example, EDMA3 channel 25 is used and TCC is also set to 25. The TCINTEN may also be set to trigger interrupt 25 when the last 1 Kbyte array is transferred. The CPU starts the EDMA3 transfer by writing to the appropriate bit of the event set register (ESR.E25). The EDMA3 transfers the first 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 TCC field. This intermediate transfer completion chaining event causes EDMA3 channel 25 to transfer the next 1 Kbyte array. This process continues until the transfer parameters are exhausted, at which point the EDMA3 has completed the 16K byte transfer. This method breaks up a large transfer into smaller packets, thus providing natural time slices in the transfer such that other events may be processed. [Figure 3-20](#) shows the EDMA3 setup and illustration of the broken up smaller packet transfers.

Figure 3-20. Smaller Packet Data Transfers Example



Registers

4.1 Register Memory Maps

For the register memory maps, see the *TMS320C6457 Fixed-Point Digital Signal Processor* data manual ([SPRS582](#)).

4.2 EDMA3 Channel Controller Control Registers

[Table 4-1](#) lists the memory-mapped registers for the EDMA3 channel controller (EDMACC). For the memory address of these registers and for the shadow region addresses, see the device-specific data manual. All other register offset addresses not listed in [Table 4-1](#) should be considered as reserved locations and the register contents should not be modified.

Table 4-1. EDMACC Registers

| Offset | Acronym | Register Description | See |
|-------------|------------|------------------------------------|---------------------------------|
| 0000h | PID | Peripheral Identification Register | Section 4.2.1.1 |
| 0004h | CCCFG | EDMA3CC Configuration Register | Section 4.2.1.2 |
| 0100h-01FCh | DCHMAP0-63 | DMA Channel 0-63 Mapping Registers | Section 4.2.1.3 |
| 0200h | QCHMAP0 | QDMA Channel 0 Mapping Register | Section 4.2.1.4 |
| 0204h | QCHMAP1 | QDMA Channel 1 Mapping Register | Section 4.2.1.4 |
| 0208h | QCHMAP2 | QDMA Channel 2 Mapping Register | Section 4.2.1.4 |
| 020Ch | QCHMAP3 | QDMA Channel 3 Mapping Register | Section 4.2.1.4 |
| 0210h | QCHMAP4 | QDMA Channel 4 Mapping Register | Section 4.2.1.4 |
| 0214h | QCHMAP5 | QDMA Channel 5 Mapping Register | Section 4.2.1.4 |
| 0218h | QCHMAP6 | QDMA Channel 6 Mapping Register | Section 4.2.1.4 |
| 021Ch | QCHMAP7 | QDMA Channel 7 Mapping Register | Section 4.2.1.4 |
| 0240h | DMAQNUM0 | DMA Queue Number Register 0 | Section 4.2.1.5 |
| 0244h | DMAQNUM1 | DMA Queue Number Register 1 | Section 4.2.1.5 |
| 0248h | DMAQNUM2 | DMA Queue Number Register 2 | Section 4.2.1.5 |
| 024Ch | DMAQNUM3 | DMA Queue Number Register 3 | Section 4.2.1.5 |
| 0250h | DMAQNUM4 | DMA Queue Number Register 4 | Section 4.2.1.5 |
| 0254h | DMAQNUM5 | DMA Queue Number Register 5 | Section 4.2.1.5 |
| 0258h | DMAQNUM6 | DMA Queue Number Register 6 | Section 4.2.1.5 |
| 025Ch | DMAQNUM7 | DMA Queue Number Register 7 | Section 4.2.1.5 |
| 0260h | QDMAQNUM | QDMA Queue Number Register | Section 4.2.1.6 |
| 0280h | QUETCMAP | Queue-to-TC Mapping Register | Section 4.2.1.7 |
| 0284h | QUEPRI | Queue Priority Register | Section 4.2.1.8 |
| 0300h | EMR | Event Missed Register | Section 4.2.2.1 |
| 0304h | EMRH | Event Missed Register High | Section 4.2.2.1 |
| 0308h | EMCR | Event Missed Clear Register | Section 4.2.2.2 |
| 030Ch | EMCRH | Event Missed Clear Register High | Section 4.2.2.2 |
| 0310h | QEMR | QDMA Event Missed Register | Section 4.2.2.3 |
| 0314h | QEMCR | QDMA Event Missed Clear Register | Section 4.2.2.4 |

Table 4-1. EDMACC Registers (continued)

| Offset | Acronym | Register Description | See |
|-------------|------------|---|---------------------------------|
| 0318h | CCERR | EDMA3CC Error Register | Section 4.2.2.5 |
| 031Ch | CCERRCLR | EDMA3CC Error Clear Register | Section 4.2.2.6 |
| 0320h | EEVAL | Error Evaluate Register | Section 4.2.2.7 |
| 0340h | DRAE0 | DMA Region Access Enable Register for Region 0 | Section 4.2.3.1 |
| 0344h | DRAEH0 | DMA Region Access Enable Register High for Region 0 | Section 4.2.3.1 |
| 0348h | DRAE1 | DMA Region Access Enable Register for Region 1 | Section 4.2.3.1 |
| 034Ch | DRAEH1 | DMA Region Access Enable Register High for Region 1 | Section 4.2.3.1 |
| 0350h | DRAE2 | DMA Region Access Enable Register for Region 2 | Section 4.2.3.1 |
| 0354h | DRAEH2 | DMA Region Access Enable Register High for Region 2 | Section 4.2.3.1 |
| 0358h | DRAE3 | DMA Region Access Enable Register for Region 3 | Section 4.2.3.1 |
| 035Ch | DRAEH3 | DMA Region Access Enable Register High for Region 3 | Section 4.2.3.1 |
| 0360h | DRAE4 | DMA Region Access Enable Register for Region 4 | Section 4.2.3.1 |
| 0364h | DRAEH4 | DMA Region Access Enable Register High for Region 4 | Section 4.2.3.1 |
| 0368h | DRAE5 | DMA Region Access Enable Register for Region 5 | Section 4.2.3.1 |
| 036Ch | DRAEH5 | DMA Region Access Enable Register High for Region 5 | Section 4.2.3.1 |
| 0370h | DRAE6 | DMA Region Access Enable Register for Region 6 | Section 4.2.3.1 |
| 0374h | DRAEH6 | DMA Region Access Enable Register High for Region 6 | Section 4.2.3.1 |
| 0378h | DRAE7 | DMA Region Access Enable Register for Region 7 | Section 4.2.3.1 |
| 037Ch | DRAEH7 | DMA Region Access Enable Register High for Region 7 | Section 4.2.3.1 |
| 0380h-039Ch | QRAE0-7 | QDMA Region Access Enable Registers for Region 0-7 | Section 4.2.3.2 |
| 0400h-0504h | QOE0-Q5E15 | Event Queue Entry Registers Q0E0-Q5E15 | Section 4.2.4.1 |
| 0600h-0614h | QSTAT0-5 | Queue Status Registers 0-5 | Section 4.2.4.2 |
| 0620h | QWMTHRA | Queue Watermark Threshold A Register | Section 4.2.4.3 |
| 0624h | QWMTHRB | Queue Watermark Threshold B Register | Section 4.2.4.4 |
| 0640h | CCSTAT | EDMA3CC Status Register | Section 4.2.4.5 |
| 0800h | MPFAR | Memory Protection Fault Address Register | Section 4.2.5.1 |
| 0804h | MPFSR | Memory Protection Fault Status Register | Section 4.2.5.2 |
| 0808h | MPFCR | Memory Protection Fault Command Register | Section 4.2.5.3 |
| 080Ch | MPPAG | Memory Protection Page Attribute Global Register | Section 4.2.5.4 |
| 0810h-082Ch | MPPA0-7 | Memory Protection Page Attribute Registers 0-7 | Section 4.2.5.4 |
| 1000h | ER | Event Register | Section 4.2.6.1 |
| 1004h | ERH | Event Register High | Section 4.2.6.1 |
| 1008h | ECR | Event Clear Register | Section 4.2.6.2 |
| 100Ch | ECRH | Event Clear Register High | Section 4.2.6.2 |
| 1010h | ESR | Event Set Register | Section 4.2.6.3 |
| 1014h | ESRH | Event Set Register High | Section 4.2.6.3 |
| 1018h | CER | Chained Event Register | Section 4.2.6.4 |
| 101Ch | CERH | Chained Event Register High | Section 4.2.6.4 |
| 1020h | EER | Event Enable Register | Section 4.2.6.5 |
| 1024h | EERH | Event Enable Register High | Section 4.2.6.5 |
| 1028h | EECR | Event Enable Clear Register | Section 4.2.6.6 |
| 102Ch | EECRH | Event Enable Clear Register High | Section 4.2.6.6 |
| 1030h | EESR | Event Enable Set Register | Section 4.2.6.7 |
| 1034h | EESRH | Event Enable Set Register High | Section 4.2.6.7 |
| 1038h | SER | Secondary Event Register | Section 4.2.6.8 |
| 103Ch | SERH | Secondary Event Register High | Section 4.2.6.8 |

Table 4-1. EDMACC Registers (continued)

| Offset | Acronym | Register Description | See |
|--|----------------|--------------------------------------|---------------------------------|
| 1040h | SECR | Secondary Event Clear Register | Section 4.2.6.9 |
| 1044h | SECRH | Secondary Event Clear Register High | Section 4.2.6.9 |
| 1050h | IER | Interrupt Enable Register | Section 4.2.7.1 |
| 1054h | IERH | Interrupt Enable Register High | Section 4.2.7.1 |
| 1058h | IECR | Interrupt Enable Clear Register | Section 4.2.7.2 |
| 105Ch | IECRH | Interrupt Enable Clear Register High | Section 4.2.7.2 |
| 1060h | IESR | Interrupt Enable Set Register | Section 4.2.7.3 |
| 1064h | IESRH | Interrupt Enable Set Register High | Section 4.2.7.3 |
| 1068h | IPR | Interrupt Pending Register | Section 4.2.7.4 |
| 106Ch | IPRH | Interrupt Pending Register High | Section 4.2.7.4 |
| 1070h | ICR | Interrupt Clear Register | Section 4.2.7.5 |
| 1074h | ICRH | Interrupt Clear Register High | Section 4.2.7.5 |
| 1078h | IEVAL | Interrupt Evaluate Register | Section 4.2.7.6 |
| 1080h | QER | QDMA Event Register | Section 4.2.8.1 |
| 1084h | QEER | QDMA Event Enable Register | Section 4.2.8.2 |
| 1088h | QEECR | QDMA Event Enable Clear Register | Section 4.2.8.3 |
| 108Ch | QEESR | QDMA Event Enable Set Register | Section 4.2.8.4 |
| 1090h | QSER | QDMA Secondary Event Register | Section 4.2.8.5 |
| 1094h | QSECR | QDMA Secondary Event Clear Register | Section 4.2.8.6 |
| Shadow Region 0 Channel Registers | | | |
| 2000h | ER | Event Register | Section 4.2.6.1 |
| 2004h | ERH | Event Register High | Section 4.2.6.1 |
| 2008h | ECR | Event Clear Register | Section 4.2.6.2 |
| 200Ch | ECRH | Event Clear Register High | Section 4.2.6.2 |
| 2010h | ESR | Event Set Register | Section 4.2.6.3 |
| 2014h | ESRH | Event Set Register High | Section 4.2.6.3 |
| 2018h | CER | Chained Event Register | Section 4.2.6.4 |
| 201Ch | CERH | Chained Event Register High | Section 4.2.6.4 |
| 2020h | EER | Event Enable Register | Section 4.2.6.5 |
| 2024h | EERH | Event Enable Register High | Section 4.2.6.5 |
| 2028h | EECR | Event Enable Clear Register | Section 4.2.6.6 |
| 202Ch | EECRH | Event Enable Clear Register High | Section 4.2.6.6 |
| 2030h | EESR | Event Enable Set Register | Section 4.2.6.7 |
| 2034h | EESRH | Event Enable Set Register High | Section 4.2.6.7 |
| 2038h | SER | Secondary Event Register | Section 4.2.6.8 |
| 203Ch | SERH | Secondary Event Register High | Section 4.2.6.8 |
| 2040h | SECR | Secondary Event Clear Register | Section 4.2.6.9 |
| 2044h | SECRH | Secondary Event Clear Register High | Section 4.2.6.9 |
| 2050h | IER | Interrupt Enable Register | Section 4.2.7.1 |
| 2054h | IERH | Interrupt Enable Register High | Section 4.2.7.1 |
| 2058h | IECR | Interrupt Enable Clear Register | Section 4.2.7.2 |
| 205Ch | IECRH | Interrupt Enable Clear Register High | Section 4.2.7.2 |
| 2060h | IESR | Interrupt Enable Set Register | Section 4.2.7.3 |
| 2064h | IESRH | Interrupt Enable Set Register High | Section 4.2.7.3 |
| 2068h | IPR | Interrupt Pending Register | Section 4.2.7.4 |
| 206Ch | IPRH | Interrupt Pending Register High | Section 4.2.7.4 |

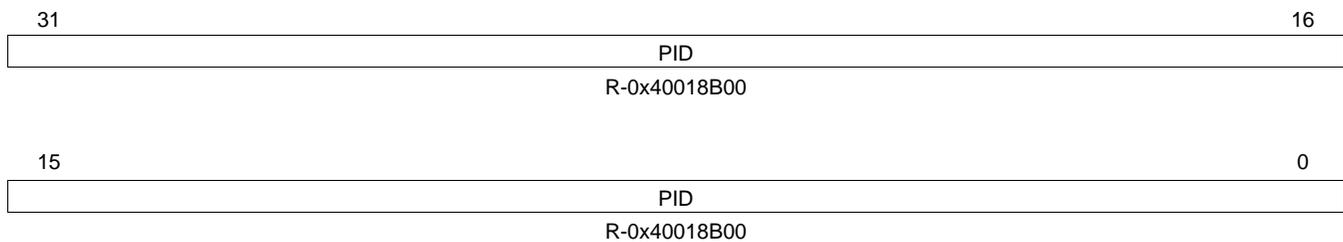
Table 4-1. EDMACC Registers (continued)

| Offset | Acronym | Register Description | See |
|-------------|---------|---|---------------------------------|
| 2070h | ICR | Interrupt Clear Register | Section 4.2.7.5 |
| 2074h | ICRH | Interrupt Clear Register High | Section 4.2.7.5 |
| 2078h | IEVAL | Interrupt Evaluate Register | Section 4.2.7.6 |
| 2080h | QER | QDMA Event Register | Section 4.2.8.1 |
| 2084h | QEER | QDMA Event Enable Register | Section 4.2.8.2 |
| 2088h | QEECR | QDMA Event Enable Clear Register | Section 4.2.8.3 |
| 208Ch | QEESR | QDMA Event Enable Set Register | Section 4.2.8.4 |
| 2090h | QSER | QDMA Secondary Event Register | Section 4.2.8.5 |
| 2094h | QSECR | QDMA Secondary Event Clear Register | Section 4.2.8.6 |
| 2200h-2294h | - | Shadow Region 1 Channel Registers | |
| 2400h-2494h | - | Shadow Region 2 Channel Registers | |
| ... | | ... | |
| 2E00h-2E94h | - | Shadow Channel Registers for MP Space 7 | |

4.2.1 Global Registers

4.2.1.1 Peripheral Identification Register (PID)

The peripheral identification register (PID) uniquely identifies the EDMA3CC and the specific revision of the EDMA3CC. The PID is shown in [Figure 4-1](#) and described in [Table 4-2](#).

Figure 4-1. Peripheral ID Register (PID)


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

Table 4-2. Peripheral ID Register (PID) Field Descriptions

| Bit | Field | Value | Description |
|------|-------|--------------|---|
| 31-0 | PID | 0-FFFF FFFFh | Peripheral identifier uniquely identifies the EDMA3CC and the specific revision of the EDMA3CC. |

4.2.1.2 EDMA3CC Configuration Register (CCCFG)

The EDMA3CC configuration register (CCCFG) provides the features/resources for the EDMA3CC in a particular device. The CCCFG is shown in [Figure 4-2](#) and described in [Table 4-3](#).

Figure 4-2. EDMA3CC Configuration Register (CCCFG)

| | | | | | | | |
|----------|-------------|----------|----|----------|-----------|----------|-------------|
| 31 | | | | | 26 | 25 | 24 |
| Reserved | | | | | | MP_EXIST | CHMAP_EXIST |
| R-x | | | | | | R-1 | R-1 |
| 23 | 22 | 21 | 20 | 19 | 18 | | |
| Reserved | | NUM_REGN | | Reserved | NUM_EVQUE | | |
| R-0 | | R-3h | | R-x | R-5h | | |
| 15 | 14 | 12 | | 11 | 10 | 8 | |
| Reserved | NUM_PAENTRY | | | Reserved | NUM_INTCH | | |
| R-x | R-4h | | | R-x | R-4h | | |
| 7 | 6 | 4 | | 3 | 2 | 0 | |
| Reserved | NUM_QDMACH | | | Reserved | NUM_DMACH | | |
| R-x | R-4h | | | R-x | R-5h | | |

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

Table 4-3. EDMA3CC Configuration Register (CCCFG) Field Descriptions

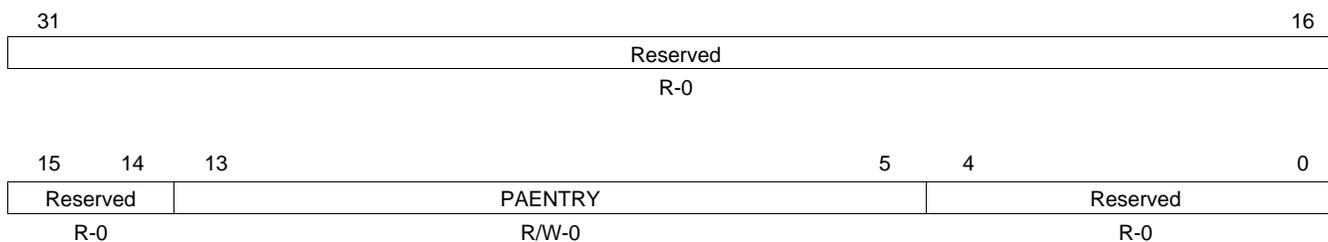
| Bit | Field | Value | Description |
|-------|-------------|-------|--|
| 31-26 | Reserved | 0 | Reserved |
| 25 | MP_EXIST | 0 | Memory protection existence. Reserved |
| | | 1 | Memory protection logic included |
| 24 | CHMAP_EXIST | 0 | Channel mapping existence. Reserved |
| | | 1 | Channel mapping logic included |
| 23-22 | Reserved | 0 | Reserved |
| 21-20 | NUM_REGN | 0-3h | Number of MP and shadow regions. |
| | | 0-2h | Reserved |
| | | 3h | 8 regions |
| 19 | Reserved | 0 | Reserved |
| 18-16 | NUM_EVQUE | 0-7h | Number of queues/number of TCs. |
| | | 0-4h | Reserved |
| | | 5h | 6 EDMA3TCs/Event Queues |
| | | 6h-7h | Reserved |
| 15 | Reserved | 0 | Reserved |
| 14-12 | NUM_PAENTRY | 0-7h | Number of PaRAM sets. |
| | | 0-3h | Reserved |
| | | 4h | 256 PaRAM sets |
| | | 5h-7h | Reserved |
| 11 | Reserved | 0 | Reserved |

Table 4-3. EDMA3CC Configuration Register (CCCFG) Field Descriptions (continued)

| Bit | Field | Value | Description |
|------|------------|-----------------------------|--|
| 10-8 | NUM_INTCH | 0-7h 0-3h 4h 5h-7h | Number of interrupt channels. Reserved 64 interrupt channels Reserved |
| 7 | Reserved | 0 | Reserved |
| 6-4 | NUM_QDMACH | 0-7h 0-3h 4h 5h-7h | Number of QDMA channels. Reserved 8 QDMA channels Reserved |
| 3 | Reserved | 0 | Reserved |
| 2-0 | NUM_DMACH | 0-7h 0-4h 5h 6h-7h | Number of DMA channels. Reserved 64 DMA channels Reserved |

4.2.1.3 DMA Channel Map n Registers (DCHMAP n)

The DMA channel map n register (DCHMAP n) is shown in [Figure 4-3](#) and described in [Table 4-4](#).

Figure 4-3. DMA Channel Map n Registers (DCHMAP n)


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

Table 4-4. DMA Channel Map n Registers (DCHMAP n) Field Descriptions

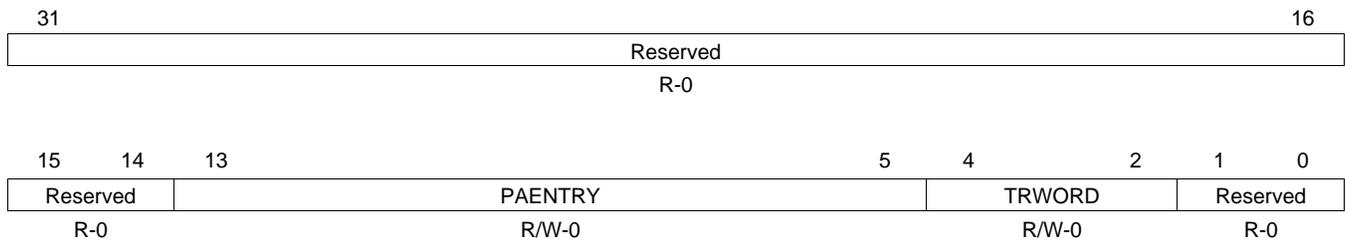
| Bit | Field | Value | Description |
|-------|----------|--------|--|
| 31-14 | Reserved | 0 | Reserved |
| 13-5 | PAENTRY | 0-1FFh | Points to the PaRAM set number for DMA channel n . |
| 4-0 | Reserved | 0 | Reserved |

4.2.1.4 QDMA Channel Map *n* Registers (QCHMAP_{*n*})

Each QDMA channel in EDMA3CC can be associated with any PaRAM set available on the device. Furthermore, the specific trigger word (0-7) of the PaRAM set can be programmed. The PaRAM set association and trigger word for every QDMA channel register is configurable using the QDMA channel map *n* register (QCHMAP_{*n*}). The QCHMAP_{*n*} is shown in Figure 4-4 and described in Table 4-5.

Note: At reset the QDMA channel map registers for all QDMA channels point to PaRAM set 0. If an application makes use of both a DMA channel that points to PaRAM set 0 and any QDMA channels, ensure that QCHMAP_{*n*} is programmed appropriately to point to a different PaRAM entry.

Figure 4-4. QDMA Channel Map *n* Registers (QCHMAP_{*n*})



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

Table 4-5. QDMA Channel Map *n* Registers (QCHMAP_{*n*}) Field Descriptions

| Bit | Field | Value | Description |
|-------|----------|------------------------------|--|
| 31-14 | Reserved | 0 | Reserved |
| 13-5 | PAENTRY | 0-1FFh 0-FFh 100h-1FFh | PAENTRY points to the PaRAM set number for QDMA channel <i>n</i> . Parameter entry 0 through 255 Reserved |
| 4-2 | TRWORD | 0-7h | Points to the specific trigger word of the PaRAM set defined by PAENTRY. A write to the trigger word results in a QDMA event being recognized. |
| 1-0 | Reserved | 0 | Reserved |

4.2.1.5 DMA Channel Queue n Number Registers (DMAQNUM n)

The DMA channel queue number register (DMAQNUM n) allows programmability of each of the 64 DMA channels in the EDMA3CC to submit its associated synchronization event to any event queue in the EDMA3CC. At reset, all channels point to event queue 0. The DMAQNUM n is shown in Figure 4-5 and described in Table 4-6. Table 4-7 shows the channels and their corresponding bits in DMAQNUM n .

Note: Because the event queues in EDMA3CC have a fixed association to the transfer controllers, that is, Q0 TRs are submitted to TC0, Q1 TRs are submitted to TC1, etc., by programming DMAQNUM n for a particular DMA channel n also dictates which transfer controller is utilized for the data movement (or which EDMA3TC receives the TR request).

Figure 4-5. DMA Channel Queue n Number Registers (DMAQNUM n)

| | | | | | | | | | | | |
|------|-------|----|------|-------|----|------|-------|----|------|-------|----|
| 31 | 30 | 28 | 27 | 26 | 24 | 23 | 22 | 20 | 19 | 18 | 16 |
| Rsvd | En | | Rsvd | En | | Rsvd | En | | Rsvd | En | |
| R-0 | R/W-0 | | R-0 | R/W-0 | | R-0 | R/W-0 | | R-0 | R/W-0 | |
| 15 | 14 | 12 | 11 | 10 | 8 | 7 | 6 | 4 | 3 | 2 | 0 |
| Rsvd | En | | Rsvd | En | | Rsvd | En | | Rsvd | En | |
| R-0 | R/W-0 | | R-0 | R/W-0 | | R-0 | R/W-0 | | R-0 | R/W-0 | |

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

Table 4-6. DMA Channel Queue n Number Registers (DMAQNUM n) Field Descriptions

| Bit | Field | Value | Description |
|------|-------|-------|---|
| 31-0 | En | 0-7h | DMA queue number. Contains the event queue number to be used for the corresponding DMA channel. Programming DMAQNUM n for an event queue number to a value more than the number of queues available in the EDMA3CC results in undefined behavior. |
| | | 0 | Event n is queued on Q0. |
| | | 1h | Event n is queued on Q1. |
| | | 2h | Event n is queued on Q2. |
| | | 3h | Event n is queued on Q3. |
| | | 4h | Event n is queued on Q4. |
| | | 5h | Event n is queued on Q5. |
| | | 6h-7h | Reserved |

Table 4-7. Bits in DMAQNUM n

| En bit | Channel Number (DMAQNUM n) | | | | | | | |
|--------|-------------------------------|-----|-----|-----|-----|-----|-----|-----|
| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 0-2 | E0 | E8 | E16 | E24 | E32 | E40 | E48 | E56 |
| 4-6 | E1 | E9 | E17 | E25 | E33 | E41 | E49 | E57 |
| 8-10 | E2 | E10 | E18 | E26 | E34 | E42 | E50 | E58 |
| 12-14 | E3 | E11 | E19 | E27 | E35 | E43 | E51 | E59 |
| 16-18 | E4 | E12 | E20 | E28 | E36 | E44 | E52 | E60 |
| 20-22 | E5 | E13 | E21 | E29 | E37 | E45 | E53 | E61 |
| 24-26 | E6 | E14 | E22 | E30 | E38 | E46 | E54 | E62 |
| 28-30 | E7 | E15 | E23 | E31 | E39 | E47 | E55 | E63 |

4.2.1.6 QDMA Channel Queue Number Register (QDMAQNUM)

The QDMA channel queue number register (QDMAQNUM) is used to program all the QDMA channels in the EDMA3CC to submit the associated QDMA event to any of the event queues in the EDMA3CC. The QDMAQNUM is shown in [Figure 4-6](#) and described in [Table 4-8](#).

Figure 4-6. QDMA Channel Queue Number Register (QDMAQNUM)

| | | | | | | | | | | | |
|------|-------|----|------|-------|----|------|-------|----|------|-------|----|
| 31 | 30 | 28 | 27 | 26 | 24 | 23 | 22 | 20 | 19 | 18 | 16 |
| Rsvd | E7 | | Rsvd | E6 | | Rsvd | E5 | | Rsvd | E4 | |
| R-0 | R/W-0 | | R-0 | R/W-0 | | R-0 | R/W-0 | | R-0 | R/W-0 | |
| 15 | 14 | 12 | 11 | 10 | 8 | 7 | 6 | 4 | 3 | 2 | 0 |
| Rsvd | E3 | | Rsvd | E2 | | Rsvd | E1 | | Rsvd | E0 | |
| R-0 | R/W-0 | | R-0 | R/W-0 | | R-0 | R/W-0 | | R-0 | R/W-0 | |

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

Table 4-8. QDMA Channel Queue Number Register (QDMAQNUM) Field Descriptions

| Bit | Field | Value | Description |
|------|-------|-------|---|
| 31-0 | E_n | 0-7h | QDMA queue number. Contains the event queue number to be used for the corresponding QDMA channel. |
| | | 0 | Event <i>n</i> is queued on Q0. |
| | | 1h | Event <i>n</i> is queued on Q1. |
| | | 2h | Event <i>n</i> is queued on Q2. |
| | | 3h | Event <i>n</i> is queued on Q3. |
| | | 4h | Event <i>n</i> is queued on Q4. |
| | | 5h | Event <i>n</i> is queued on Q5. |
| | | 6h-7h | Reserved |

4.2.1.7 Queue-to-TC Mapping Register (QUETCMAP)

The Queue-to-TC Mapping Register (QUETCMAP) is shown in [Figure 4-7](#) and described in [Table 4-9](#).

Figure 4-7. Queue-to-TC Mapping Register (QUETCMAP)

| | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|----------|---------|----|----|------|---------|---------|----|------|---------|---------|----|------|---------|---|----|--|---|--|--|---|---|--|--|---|--|--|---|
| 31 | | | | | | 23 | 22 | | | 20 | 19 | 18 | | | 16 | | | | | | | | | | | | |
| Reserved | | | | | | TCNUMQ5 | | | Rsvd | TCNUMQ4 | | | | | | | | | | | | | | | | | |
| R-0 | | | | | | R/W-5 | | | R-0 | R/W-4 | | | | | | | | | | | | | | | | | |
| | | 15 | 14 | | | 12 | 11 | | | 10 | | | 8 | 7 | | | 6 | | | 4 | 3 | | | 2 | | | 0 |
| Rsvd | TCNUMQ3 | | | Rsvd | TCNUMQ2 | | | Rsvd | TCNUMQ1 | | | Rsvd | TCNUMQ0 | | | | | | | | | | | | | | |
| R-0 | R/W-3 | | | R-0 | R/W-2 | | | R-0 | R/W-1 | | | R-0 | R/W-0 | | | | | | | | | | | | | | |

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

Table 4-9. Queue-to-TC Mapping Register (QUETCMAP) Field Descriptions

| Bit | Field | Value | Description |
|-------------------------------------|------------|-------|---|
| 31-23, 19, 15, 11, 7, 3 | Reserved | | Reserved |
| 22-20, 18-16, 14-12, 10-8, 6-4, 2-0 | TCNUMQ n | | TC number for queue n . Defines the TC number to which Event Queue n TRs are written via the TR bus interface. Only used if TCs are not identical (e.g., TC0 FIFO is 32 bytes while TC1 FIFO is 64 bytes). Note that the same TC cannot be assigned to multiple queues (i.e., TCNUMQ n \neq TCNUMQ m). |

4.2.1.8 Queue Priority Register (QUEPRI)

The queue priority register (QUEPRI) allows you to change the priority of the individual queues and, therefore, the priority of the transfer request (TR) associated with the events queued in the queue. Because the queue-to-TC mapping is available, programming QUEPRI essentially governs the priority of the associated transfer controller(s) read/write commands with respect to the other bus masters in the device. You can modify the EDMA3TC priority to obtain the desired system performance. The QUEPRI is shown in [Figure 4-8](#) and described in [Table 4-10](#).

Figure 4-8. Queue Priority Register (QUEPRI)

| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|----------|-------|--|------|-------|--|------|-------|--|------|-------|----|------|-------|----|----|----|----|--|---|----|---|--|--|---|---|---|--|--|---|
| 31 | | | | | | | | | | | 23 | 22 | | | 20 | 19 | 18 | | | 16 | | | | | | | | | |
| Reserved | | | | | | | | | | PRIQ5 | | Rsvd | PRIQ4 | | | | | | | | | | | | | | | | |
| R-0 | | | | | | | | | | R/W-0 | | R-0 | R/W-0 | | | | | | | | | | | | | | | | |
| | | | | | | | | | | 15 | 14 | | | 12 | 11 | 10 | | | 8 | 7 | 6 | | | 4 | 3 | 2 | | | 0 |
| Rsvd | PRIQ3 | | Rsvd | PRIQ2 | | Rsvd | PRIQ1 | | Rsvd | PRIQ0 | | | | | | | | | | | | | | | | | | | |
| R-0 | R/W-0 | | R-0 | R/W-0 | | R-0 | R/W-0 | | R-0 | R/W-0 | | | | | | | | | | | | | | | | | | | |

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

Table 4-10. Queue Priority Register (QUEPRI) Field Descriptions

| Bit | Field | Value | Description |
|-------|----------|-------|--|
| 31-23 | Reserved | 0 | Reserved |
| 22-20 | PRIQ5 | 0-7h | Priority level for queue 5. Dictates the priority level used by TC5 relative to other masters in the device. A value of 0 means highest priority and a value of 7 means lowest priority. |
| 19 | Reserved | 0 | Reserved |
| 18-16 | PRIQ4 | 0-7h | Priority level for queue 4. Dictates the priority level used by TC4 relative to other masters in the device. A value of 0 means highest priority and a value of 7 means lowest priority. |
| 15 | Reserved | 0 | Reserved |
| 14-12 | PRIQ3 | 0-7h | Priority level for queue 3. Dictates the priority level used by TC3 relative to other masters in the device. A value of 0 means highest priority and a value of 7 means lowest priority. |
| 11 | Reserved | 0 | Reserved |
| 10-8 | PRIQ2 | 0-7h | Priority level for queue 2. Dictates the priority level used by TC2 relative to other masters in the device. A value of 0 means highest priority and a value of 7 means lowest priority. |
| 7 | Reserved | 0 | Reserved |
| 6-4 | PRIQ1 | 0-7h | Priority level for queue 1. Dictates the priority level used by TC1 relative to other masters in the device. A value of 0 means highest priority and a value of 7 means lowest priority. |
| 3 | Reserved | 0-1h | Reserved |
| 2-0 | PRIQ0 | 0-7h | Priority level for queue 0. Dictates the priority level used by TC0 relative to other masters in the device. A value of 0 means highest priority and a value of 7 means lowest priority. |

4.2.2 Error Registers

The EDMA3CC contains a set of registers that provide information on missed DMA and/or QDMA events, and instances when event queue thresholds are exceeded. If any of the bits in these registers is set, it results in the EDMA3CC generating an error interrupt.

4.2.2.1 Event Missed Registers (EMR/EMRH)

For a particular DMA channel, if a second event is received prior to the first event getting cleared/serviced, the bit corresponding to that channel is set/asserted in the event missed registers (EMR/EMRH). All trigger types are treated individually, that is, manual triggered (ESR/ESRH), chain triggered (CER/CERH), and event triggered (ER/ERH) are all treated separately. The EMR/EMRH bits for a channel are also set if an event on that channel encounters a NULL entry (or a NULL TR is serviced). If any EMR/EMRH bit is set (and all errors, including bits in other error registers (QEMR, CCERR) were previously cleared), the EDMA3CC generates an error interrupt. For details on EDMA3CC error interrupt generation, see [Section 2.9.4](#).

The EMR is shown in [Figure 4-9](#) and described in [Table 4-11](#). The EMRH is shown in [Figure 4-10](#) and described in [Table 4-12](#).

Figure 4-9. Event Missed Register (EMR)

| | | | | | | | | | | | | | | | |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| E31 | E30 | E29 | E28 | E27 | E26 | E25 | E24 | E23 | E22 | E21 | E20 | E19 | E18 | E17 | E16 |
| R-0 |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| E15 | E14 | E13 | E12 | E11 | E10 | E9 | E8 | E7 | E6 | E5 | E4 | E3 | E2 | E1 | E0 |
| R-0 |

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

Table 4-11. Event Missed Register (EMR) Field Descriptions

| Bit | Field | Value | Description |
|------|-------|-------|--|
| 31-0 | E_n | | Channel 0-31 event missed. E_n is cleared by writing a 1 to the corresponding bit in the event missed clear register (EMCR). |
| | | 0 | No missed event. |
| | | 1 | Missed event occurred. |

Figure 4-10. Event Missed Register High (EMRH)

| | | | | | | | | | | | | | | | |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| E63 | E62 | E61 | E60 | E59 | E58 | E57 | E56 | E55 | E54 | E53 | E52 | E51 | E50 | E49 | E48 |
| R-0 |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| E47 | E46 | E45 | E44 | E43 | E42 | E41 | E40 | E39 | E38 | E37 | E36 | E35 | E34 | E33 | E32 |
| R-0 |

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

Table 4-12. Event Missed Register High (EMRH) Field Descriptions

| Bit | Field | Value | Description |
|------|-------|-------|---|
| 31-0 | E_n | 0 | Channel 32-63 event missed. E_n is cleared by writing a 1 to the corresponding bit in the event missed clear register high (EMCRH). No missed event. |
| | | 1 | Missed event occurred. |

4.2.2.2 Event Missed Clear Registers (EMCR/EMCRH)

Once a missed event is posted in the event missed registers (EMR/EMRH), the bit remains set and you need to clear the set bit(s). This is done by way of CPU writes to the event missed clear registers (EMCR/EMCRH). Writing a 1 to any of the bits clears the corresponding missed event (bit) in EMR/EMRH; writing a 0 has no effect.

The EMCR is shown in [Figure 4-11](#) and described in [Table 4-13](#). The EMCRH is shown in [Figure 4-12](#) and described in [Table 4-14](#).

Figure 4-11. Event Missed Clear Register (EMCR)

| | | | | | | | | | | | | | | | |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| E31 | E30 | E29 | E28 | E27 | E26 | E25 | E24 | E23 | E22 | E21 | E20 | E19 | E18 | E17 | E16 |
| W-0 |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| E15 | E14 | E13 | E12 | E11 | E10 | E9 | E8 | E7 | E6 | E5 | E4 | E3 | E2 | E1 | E0 |
| W-0 |

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

Table 4-13. Event Missed Clear Register (EMCR) Field Descriptions

| Bit | Field | Value | Description |
|------|-------|-------|---|
| 31-0 | E_n | 0 | Event missed 0-31 clear. All error bits must be cleared before additional error interrupts are asserted by the EDMA3CC. No effect. |
| | | 1 | Corresponding missed event bit in the event missed register (EMR) is cleared ($E_n = 0$). |

Figure 4-12. Event Missed Clear Register High (EMCRH)

| | | | | | | | | | | | | | | | |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| E63 | E62 | E61 | E60 | E59 | E58 | E57 | E56 | E55 | E54 | E53 | E52 | E51 | E50 | E49 | E48 |
| W-0 |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| E47 | E46 | E45 | E44 | E43 | E42 | E41 | E40 | E39 | E38 | E37 | E36 | E35 | E34 | E33 | E32 |
| W-0 |

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

Table 4-14. Event Missed Clear Register High (EMCRH) Field Descriptions

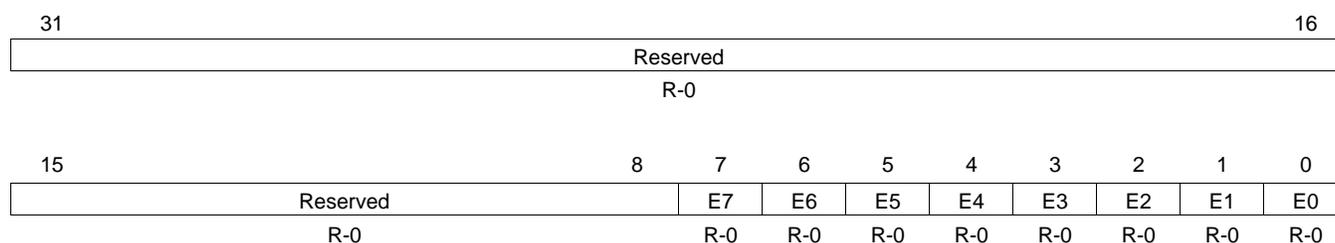
| Bit | Field | Value | Description |
|------|-------|-------|--|
| 31-0 | E_n | 0 | Event missed 32-63 clear. All error bits must be cleared before additional error interrupts are asserted by the EDMA3CC. No effect. |
| | | 1 | Corresponding missed event bit in the event missed register high (EMRH) is cleared ($E_n = 0$). |

4.2.2.3 QDMA Event Missed Register (QEMR)

For a particular QDMA channel, if two QDMA events are detected without the first event getting cleared/serviced, the bit corresponding to that channel is set/asserted in the QDMA event missed register (QEMR). The QEMR bits for a channel are also set if a QDMA event on the channel encounters a NULL entry (or a NULL TR is serviced). If any QEMR bit is set (and all errors, including bits in other error registers (EMR/EMRH, CCERR) were previously cleared), the EDMA3CC generates an error interrupt. For details on EDMA3CC error interrupt generation, see [Section 2.9.4](#).

The QEMR is shown in [Figure 4-13](#) and described in [Table 4-15](#).

Figure 4-13. QDMA Event Missed Register (QEMR)



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

Table 4-15. QDMA Event Missed Register (QEMR) Field Descriptions

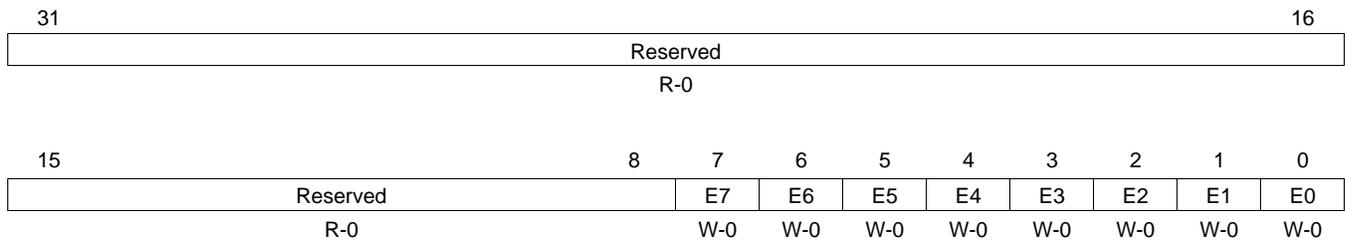
| Bit | Field | Value | Description |
|------|----------|-------|--|
| 31-8 | Reserved | 0 | Reserved |
| 7-0 | E_n | 0 | Channel 0-7 QDMA event missed. E_n is cleared by writing a 1 to the corresponding bit in the QDMA event missed clear register (QEMCR). |
| | | 0 | No missed event. |
| | | 1 | Missed event occurred. |

4.2.2.4 QDMA Event Missed Clear Register (QEMCR)

Once a missed event is posted in the QDMA event missed registers (QEMR), the bit remains set and you need to clear the set bit(s). This is done by way of CPU writes to the QDMA event missed clear registers (QEMCR). Writing a 1 to any of the bits clears the corresponding missed event (bit) in QEMR; writing a 0 has no effect.

The QEMCR is shown in [Figure 4-14](#) and described in [Table 4-16](#).

Figure 4-14. QDMA Event Missed Clear Register (QEMCR)



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

Table 4-16. QDMA Event Missed Clear Register (QEMCR) Field Descriptions

| Bit | Field | Value | Description |
|------|----------|-------|---|
| 31-8 | Reserved | 0 | Reserved |
| 7-0 | E_n | 0 | QDMA event missed clear. All error bits must be cleared before additional error interrupts are asserted by the EDMA3CC. |
| | | 0 | No effect. |
| | | 1 | Corresponding missed event bit in the QDMA event missed register (QEMR) is cleared ($E_n = 0$). |

4.2.2.5 EDMA3CC Error Register (CCERR)

The EDMA3CC error register (CCERR) indicates whether or not at any instant of time the number of events queued up in any of the event queues exceeds or equals the threshold/watermark value that is set in the queue watermark threshold register (QWMTHRA). Additionally, CCERR also indicates if when the number of outstanding TRs that have been programmed to return transfer completion code (TRs which have the TCINTEN or TCCHEN bit in OPT set) to the EDMA3CC has exceeded the maximum allowed value of 63. If any bit in CCERR is set (and all errors, including bits in other error registers (EMR/EMRH, QEMR) were previously cleared), the EDMA3CC generates an error interrupt. For details on EDMA3CC error interrupt generation, see [Section 2.9.4](#). Once the error bits are set in CCERR, they can only be cleared by writing to the corresponding bits in the EDMA3CC error clear register (CCERRCLR).

The CCERR is shown in [Figure 4-15](#) and described in [Table 4-17](#).

Figure 4-15. EDMA3CC Error Register (CCERR)

| | | | | | | | | |
|----------|---|---------|---------|---------|---------|---------|---------|----|
| 31 | | | | | | | 17 | 16 |
| Reserved | | | | | | | TCCERR | |
| R-0 | | | | | | | R-0 | |
| 15 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
| Reserved | | QTHRXC5 | QTHRXC4 | QTHRXC3 | QTHRXC2 | QTHRXC1 | QTHRXC0 | |
| R-0 | | R-0 | R-0 | R-0 | R-0 | R-0 | R-0 | |

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

Table 4-17. EDMA3CC Error Register (CCERR) Field Descriptions

| Bit | Field | Value | Description |
|-------|----------|-------|---|
| 31-17 | Reserved | 0 | Reserved |
| 16 | TCCERR | | Transfer completion code error. TCCERR is cleared by writing a 1 to the corresponding bit in the EDMA3CC error clear register (CCERRCLR). |
| | | 0 | Total number of allowed TCCs outstanding has not been reached. |
| | | 1 | Total number of allowed TCCs has been reached. |
| 15-6 | Reserved | 0 | Reserved |
| 5 | QTHRXC5 | | Queue threshold error for queue 5. QTHRXC5 is cleared by writing a 1 to the corresponding bit in the EDMA3CC error clear register (CCERRCLR). |
| | | 0 | Watermark/threshold has not been exceeded. |
| | | 1 | Watermark/threshold has been exceeded. |
| 4 | QTHRXC4 | | Queue threshold error for queue 4. QTHRXC4 is cleared by writing a 1 to the corresponding bit in the EDMA3CC error clear register (CCERRCLR). |
| | | 0 | Watermark/threshold has not been exceeded. |
| | | 1 | Watermark/threshold has been exceeded. |
| 3 | QTHRXC3 | | Queue threshold error for queue 3. QTHRXC3 is cleared by writing a 1 to the corresponding bit in the EDMA3CC error clear register (CCERRCLR). |
| | | 0 | Watermark/threshold has not been exceeded. |
| | | 1 | Watermark/threshold has been exceeded. |
| 2 | QTHRXC2 | | Queue threshold error for queue 2. QTHRXC2 is cleared by writing a 1 to the corresponding bit in the EDMA3CC error clear register (CCERRCLR). |
| | | 0 | Watermark/threshold has not been exceeded. |
| | | 1 | Watermark/threshold has been exceeded. |
| 1 | QTHRXC1 | | Queue threshold error for queue 1. QTHRXC1 is cleared by writing a 1 to the corresponding bit in the EDMA3CC error clear register (CCERRCLR). |
| | | 0 | Watermark/threshold has not been exceeded. |
| | | 1 | Watermark/threshold has been exceeded. |

Table 4-17. EDMA3CC Error Register (CCERR) Field Descriptions (continued)

| Bit | Field | Value | Description |
|-----|----------|-------|--|
| 0 | QTHRXCD0 | | Queue threshold error for queue 0. QTHRXCD0 is cleared by writing a 1 to the corresponding bit in the EDMA3CC error clear register (CCERRCLR). |
| | | 0 | Watermark/threshold has not been exceeded. |
| | | 1 | Watermark/threshold has been exceeded. |

4.2.2.6 EDMA3CC Error Clear Register (CCERRCLR)

The EDMA3CC error clear register (CCERRCLR) is used to clear any error bits that are set in the EDMA3CC error register (CCERR). In addition, CCERRCLR also clears the values of some bit fields in the queue status registers (QSTAT n) associated with a particular event queue. Writing a 1 to any of the bits clears the corresponding bit in CCERR; writing a 0 has no effect.

The CCERRCLR is shown in [Figure 4-16](#) and described in [Table 4-18](#).

Figure 4-16. EDMA3CC Error Clear Register (CCERRCLR)

| | | | | | | | | |
|----------|---|---------|---------|---------|---------|---------|---------|----|
| 31 | | | | | | | 17 | 16 |
| Reserved | | | | | | | TCCERR | |
| W-0 | | | | | | | W-0 | |
| 15 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
| Reserved | | QTHRXC5 | QTHRXC4 | QTHRXC3 | QTHRXC2 | QTHRXC1 | QTHRXC0 | |
| R-0 | | W-0 | W-0 | W-0 | W-0 | W-0 | W-0 | |

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

Table 4-18. EDMA3CC Error Clear Register (CCERRCLR) Field Descriptions

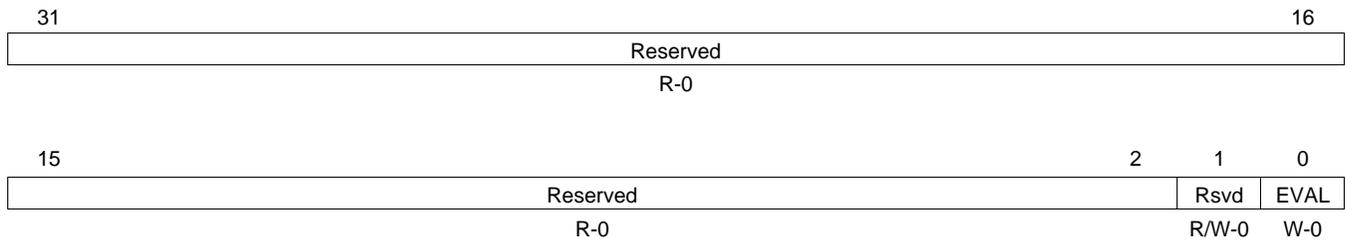
| Bit | Field | Value | Description |
|-------|----------|-------|--|
| 31-17 | Reserved | 0 | Reserved |
| 16 | TCCERR | 0 | Transfer completion code error clear No effect. |
| | | 1 | Clears the TCCERR bit in the EDMA3CC error register (CCERR). |
| 15-6 | Reserved | 0 | Reserved |
| 5 | QTHRXC5 | 0 | Queue threshold error clear for queue 5. No effect. |
| | | 1 | Clears the QTHRXC5 bit in the EDMA3CC error register (CCERR) and the WM and THRXCD bits in the queue status register 5 (QSTAT5). |
| 4 | QTHRXC4 | 0 | Queue threshold error clear for queue 4. No effect. |
| | | 1 | Clears the QTHRXC4 bit in the EDMA3CC error register (CCERR) and the WM and THRXCD bits in the queue status register 4 (QSTAT4). |
| 3 | QTHRXC3 | 0 | Queue threshold error clear for queue 3. No effect. |
| | | 1 | Clears the QTHRXC3 bit in the EDMA3CC error register (CCERR) and the WM and THRXCD bits in the queue status register 3 (QSTAT3). |
| 2 | QTHRXC2 | 0 | Queue threshold error clear for queue 2. No effect. |
| | | 1 | Clears the QTHRXC2 bit in the EDMA3CC error register (CCERR) and the WM and THRXCD bits in the queue status register 2 (QSTAT2). |
| 1 | QTHRXC1 | 0 | Queue threshold error clear for queue 1. No effect. |
| | | 1 | Clears the QTHRXC1 bit in the EDMA3CC error register (CCERR) and the WM and THRXCD bits in the queue status register 1 (QSTAT1). |
| 0 | QTHRXC0 | 0 | Queue threshold error clear for queue 0. No effect. |
| | | 1 | Clears the QTHRXC0 bit in the EDMA3CC error register (CCERR) and the WM and THRXCD bits in the queue status register 0 (QSTAT0). |

4.2.2.7 Error Evaluation Register (EEVAL)

The EDMA3CC error interrupt is asserted whenever an error bit is set in any of the error registers (EMR/EMRH, QEMR, and CCERR). For subsequent error bits that get set, the EDMA3CC error interrupt is reasserted only when transitioning from an “all the error bits cleared” to “at least one error bit is set”. Alternatively, a CPU write of 1 to the EVAL bit in the error evaluation register (EEVAL) results in reasserting the EDMA3CC error interrupt, if there are any outstanding error bits set due to subsequent error conditions. Writes of 0 have no effect.

The EEVAL is shown in [Figure 4-17](#) and described in [Table 4-19](#).

Figure 4-17. Error Evaluation Register (EEVAL)



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

Table 4-19. Error Evaluation Register (EEVAL) Field Descriptions

| Bit | Field | Value | Description |
|------|----------|-------|---|
| 31-2 | Reserved | 0 | Reserved |
| 1 | Reserved | 0 | Reserved. Always write 0 to this bit; writes of 1 to this bit are not supported and attempts to do so may result in undefined behavior. |
| 0 | EVAL | 0 | Error interrupt evaluate. No effect. |
| | | 1 | EDMA3CC error interrupt is pulsed if any errors have not been cleared in any of the error registers (EMR/EMRH, QEMR, or CCERR). |

4.2.3 Region Access Enable Registers

The region access enable register group consists of the DMA access enable registers (DRAE m and DRAEH m) and the QDMA access enable registers (QRAE m). Where m is the number of shadow regions in the EDMA3CC memory map for a device. You can configure these registers to assign ownership of DMA/QDMA channels to a particular shadow region.

4.2.3.1 DMA Region Access Enable for Region m (DRAE m)

The DMA region access enable register for shadow region m (DRAE m /DRAEH m) is programmed to allow or disallow read/write accesses on a bit-by-bit bases for all DMA registers in the shadow region m view of the DMA channel registers. For a list of all the DMA channel and interrupt registers mapped in the shadow region view, see the EDMA3CC register memory map (Table 4-1). Additionally, the DRAE m /DRAEH m configuration determines completion of which DMA channels results in assertion of the shadow region m DMA completion interrupt (see Section 2.9).

The DRAE m is shown in Figure 4-18 and described in Table 4-20. The DRAEH m is shown in Figure 4-19 and described in Table 4-20.

Figure 4-18. DMA Region Access Enable Register for Region m (DRAE m)

| | | | | | | | | | | | | | | | |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| E31 | E30 | E29 | E28 | E27 | E26 | E25 | E24 | E23 | E22 | E21 | E20 | E19 | E18 | E17 | E16 |
| RW-0 |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| E15 | E14 | E13 | E12 | E11 | E10 | E9 | E8 | E7 | E6 | E5 | E4 | E3 | E2 | E1 | E0 |
| RW-0 |

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

Figure 4-19. DMA Region Access Enable High Register for Region m (DRAEH m)

| | | | | | | | | | | | | | | | |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| E63 | E62 | E61 | E60 | E59 | E58 | E57 | E56 | E55 | E54 | E53 | E52 | E51 | E50 | E49 | E48 |
| RW-0 |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| E47 | E46 | E45 | E44 | E43 | E42 | E41 | E40 | E39 | E38 | E37 | E36 | E35 | E34 | E33 | E32 |
| RW-0 |

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

Table 4-20. DMA Region Access Enable Registers for Region M (DRAE m /DRAEH m) Field Descriptions

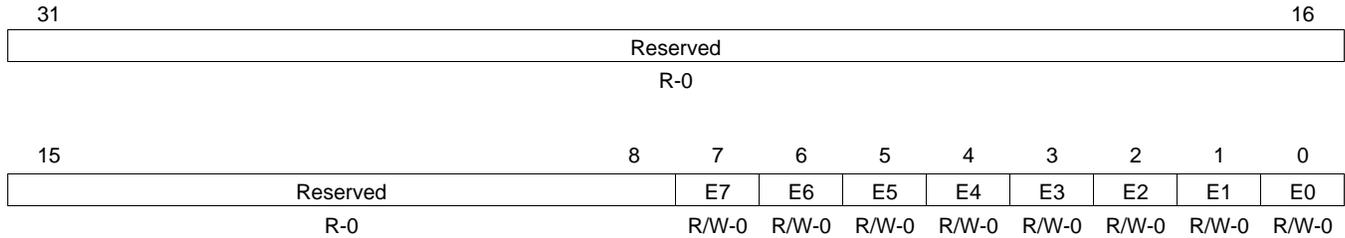
| Bit | Field | Value | Description |
|------|-------|-------|--|
| 31-0 | E n | 0 | DMA region access enable for bit n /channel n in region m . Accesses via region m address space to bit n in any DMA channel register are not allowed. Reads return 0 on bit n and writes do not modify the state of bit n . Enabled interrupt bits for bit n do not contribute to the generation of a transfer completion interrupt for shadow region m . |
| | | 1 | Accesses via region m address space to bit n in any DMA channel register are allowed. Reads return the value from bit n and writes modify the state of bit n . Enabled interrupt bits for bit n contribute to the generation of a transfer completion interrupt for shadow region m . |

4.2.3.2 QDMA Region Access Enable Registers (QRAEm)

The QDMA region access enable register for shadow region m (QRAEm) is programmed to allow or disallow read/write accesses on a bit-by-bit bases for all QDMA registers in the shadow region m view of the QDMA registers. This includes all 4-bit QDMA registers.

The QRAEm is shown in [Figure 4-20](#) and described in [Table 4-21](#).

Figure 4-20. QDMA Region Access Enable for Region m (QRAEm)



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

Table 4-21. QDMA Region Access Enable for Region M (QRAEm) Field Descriptions

| Bit | Field | Value | Description |
|------|----------|-------|---|
| 31-8 | Reserved | 0 | Reserved |
| 7-0 | E_n | 0 | QDMA region access enable for bit n /QDMA channel n in region m . Accesses via region m address space to bit n in any QDMA channel register are not allowed. Reads return 0 on bit n and writes do not modify the state of bit n . |
| | | 1 | Accesses via region m address space to bit n in any QDMA channel register are allowed. Reads return the value from bit n and writes modify the state of bit n . |

4.2.4 Status/Debug Visibility Registers

The following set of registers provide visibility into the event queues and a TR life cycle. These are useful for system debug as they provide in-depth visibility for the events queued up in the event queue and also provide information on what parts of the EDMA3CC logic are active once the event has been received by the EDMA3CC.

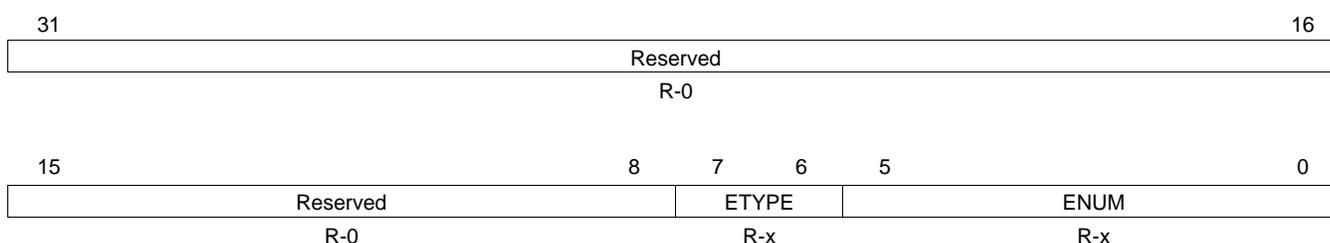
4.2.4.1 Event Queue Entry Registers (QxEy)

The event queue entry registers (QxEy) exist for all 16 queue entries (the maximum allowed queue entries) for all event queues in the EDMA3CC.

There are Q0E0 to Q0E15, Q1E0 to Q1E15, Q2E0 to Q2E15, and Q3E0 to Q3E15. Each register details the event number (ENUM) and the event type (ETYPE). For example, if the value in Q1E4 is read as 000 004Fh, this means the 4th entry in queue 1 is a manually-triggered event on DMA channel 15.

The QxEy is shown in [Figure 4-21](#) and described in [Table 4-22](#).

Figure 4-21. Event Queue Entry Registers (QxEy)



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

Table 4-22. Event Queue Entry Registers (QxEy) Field Descriptions

| Bit | Field | Value | Description |
|------|----------|-------|---|
| 31-8 | Reserved | 0 | Reserved |
| 7-6 | ETYPE | 0-3h | Event entry y in queue x. Specifies the specific event type for the given entry in the event queue. |
| | | 0 | Event triggered via ER. |
| | | 1h | Manual triggered via ESR. |
| | | 2h | Chain triggered via CER. |
| | | 3h | Autotriggered via QER. |
| 5-0 | ENUM | 0-3Fh | Event entry y in queue x. Event number: |
| | | 0-7h | QDMA channel number (0 to 7). |
| | | 0-3Fh | DMA channel/event number (0 to 63). |

4.2.4.2 Queue Status Registers (QSTAT n)

The queue status registers (QSTAT n) is shown in [Figure 4-22](#) and described in [Table 4-23](#).

Figure 4-22. Queue Status Register n (QSTAT n)

| | | | | | | | |
|----|----------|------------------------|-----|----------|-----|-------------------------|-----|
| 31 | Reserved | 25 | 24 | 23 | 21 | 20 | 16 |
| | R-0 | THRXC <small>D</small> | R-0 | Reserved | R-0 | WM | R-0 |
| 15 | Reserved | 13 | 12 | 8 | 7 | 4 | 3 |
| | R-0 | NUMVAL | R-0 | Reserved | R-0 | STRTP <small>TR</small> | R-0 |
| | | | | | | | 0 |

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

Table 4-23. Queue Status Register n (QSTAT n) Field Descriptions

| Bit | Field | Value | Description |
|-------|-------------------------|---------|--|
| 31-25 | Reserved | 0 | Reserved |
| 24 | THRXC <small>D</small> | 0 | Threshold exceeded. THRXC <small>D</small> is cleared by writing a 1 to the corresponding QTHRXC <small>D</small> n bit in the EDMA3CC error clear register (CCERRCLR). |
| | | 1 | Threshold specified by the Q n bit in the queue watermark threshold register (QWMTHRA/QWMTHRB) has not been exceeded. |
| | | 1 | Threshold specified by the Q n bit in the queue watermark threshold register (QWMTHRA/QWMTHRB) has been exceeded. |
| 23-21 | Reserved | 0 | Reserved |
| 20-16 | WM | 0-1Fh | Watermark for maximum queue usage. Watermark tracks the most entries that have been in queue n since reset or since the last time that the watermark (WM) bit was cleared. WM is cleared by writing a 1 to the corresponding QTHRXC <small>D</small> n bit in the EDMA3CC error clear register (CCERRCLR). Legal values are 0 (empty) to 10h (full). |
| | | 11h-1Fh | Reserved |
| 15-13 | Reserved | 0 | Reserved |
| 12-8 | NUMVAL | 0-1Fh | Number of valid entries in queue n . The total number of entries residing in the queue manager FIFO at a given instant. Always enabled. Legal values are 0 (empty) to 10h (full). |
| | | 11h-1Fh | Reserved |
| 7-4 | Reserved | 0 | Reserved |
| 3-0 | STRTP <small>TR</small> | 0-Fh | Start pointer. The offset to the head entry of queue n , in units of entries. Always enabled. Legal values are 0 (0th entry) to Fh (15th entry). |

4.2.4.3 Queue Watermark Threshold A Register (QWMTHRA)

The queue watermark threshold A register (QWMTHRA) is shown in [Figure 4-23](#) and described in [Table 4-24](#).

Figure 4-23. Queue Watermark Threshold A Register (QWMTHRA)

| | | | | | | | | |
|----------|----|---------|----|----------|----|---------|----|---|
| 31 | 29 | 28 | 24 | 23 | 21 | 20 | 16 | |
| Reserved | | Q3 | | Reserved | | Q2 | | |
| R-0 | | R/W-10 | | R-0 | | R/W-10 | | |
| 15 | | 13 | 12 | 8 | 7 | 5 | 4 | 0 |
| Reserved | | Q1 | | Reserved | | Q0 | | |
| R-0 | | R/W-10h | | R-0 | | R/W-10h | | |

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

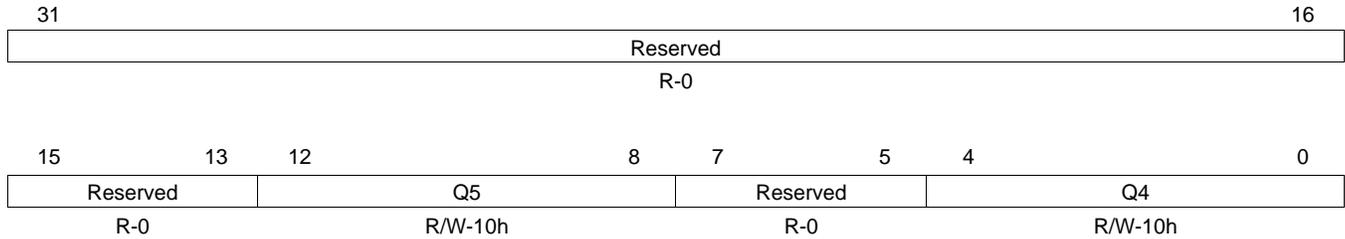
Table 4-24. Queue Watermark Threshold A Register (QWMTHRA) Field Descriptions

| Bit | Field | Value | Description |
|-------|----------|----------------------------------|--|
| 31-29 | Reserved | 0 | Reserved |
| 28-24 | Q3 | 0-1Fh 0-10h 11h 12h-1Fh | Queue threshold for queue 3 value. The QTHRCD3 bit in the EDMA3CC error register (CCERR) and the THRXCD bit in the queue status register 3 (QSTAT3) are set when the number of events in queue 3 at an instant in time (visible via the NUMVAL bit in QSTAT3) equals or exceeds the value specified by Q3. The default is 16 (maximum allowed). Disables the threshold errors. Reserved |
| 23-21 | Reserved | 0 | Reserved |
| 20-16 | Q2 | 0-1Fh 0-10h 11h 12h-1Fh | Queue threshold for queue 2 value. The QTHRCD2 bit in the EDMA3CC error register (CCERR) and the THRXCD bit in the queue status register 2 (QSTAT2) are set when the number of events in queue 2 at an instant in time (visible via the NUMVAL bit in QSTAT2) equals or exceeds the value specified by Q2. The default is 16 (maximum allowed). Disables the threshold errors. Reserved |
| 15-13 | Reserved | 0 | Reserved |
| 12-8 | Q1 | 0-1Fh 0-10h 11h 12h-1Fh | Queue threshold for queue 1 value. The QTHRCD1 bit in the EDMA3CC error register (CCERR) and the THRXCD bit in the queue status register 1 (QSTAT1) are set when the number of events in queue 1 at an instant in time (visible via the NUMVAL bit in QSTAT1) equals or exceeds the value specified by Q1. The default is 16 (maximum allowed). Disables the threshold errors. Reserved |
| 7-5 | Reserved | 0 | Reserved |
| 4-0 | Q0 | 0-1Fh 0-10h 11h 12h-1Fh | Queue threshold for queue 0 value. The QTHRCD0 bit in the EDMA3CC error register (CCERR) and the THRXCD bit in the queue status register 0 (QSTAT0) are set when the number of events in queue 0 at an instant in time (visible via the NUMVAL bit in QSTAT0) equals or exceeds the value specified by Q0. The default is 16 (maximum allowed). Disables the threshold errors. Reserved |

4.2.4.4 Queue Watermark Threshold B Register (QWMTHRB)

The queue watermark threshold B register (QWMTHRB) is shown in [Figure 4-24](#) and described in [Table 4-25](#).

Figure 4-24. Queue Watermark Threshold B Register (QWMTHRB)



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

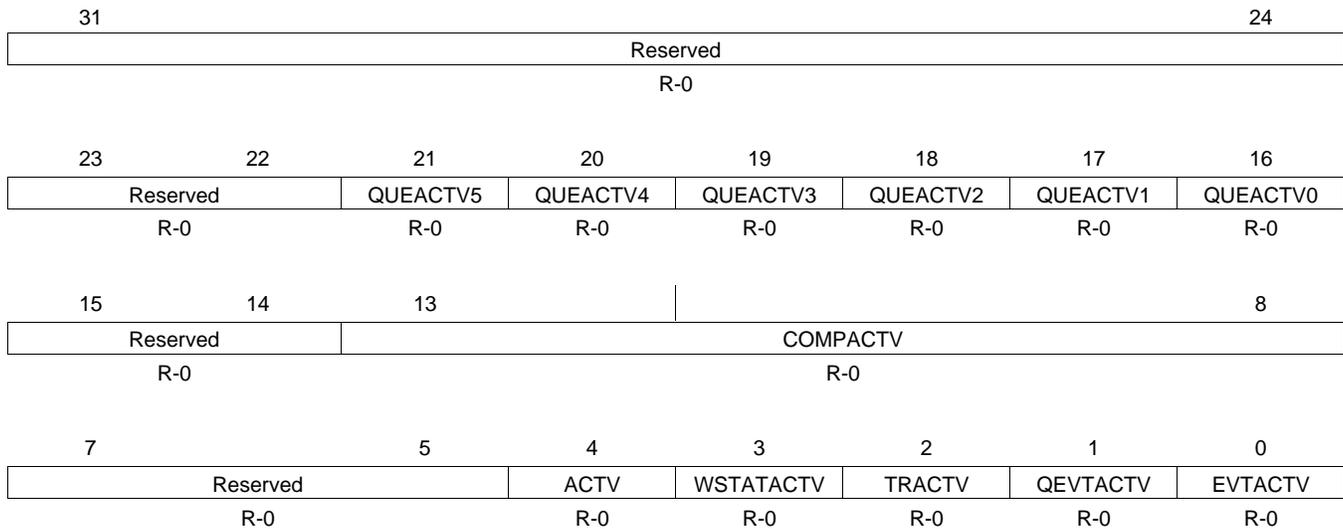
Table 4-25. Queue Watermark Threshold B Register (QWMTHRB) Field Descriptions

| Bit | Field | Value | Description |
|-------|----------|-------------------------|--|
| 31-13 | Reserved | 0 | Reserved |
| 12-8 | Q5 | 0-10h 11h 12h-1Fh | Queue threshold for queue 5 value. The QTHRCD1 bit in the EDMA3CC error register (CCERR) and the THRXCD bit in the queue status register 5 (QSTAT5) are set when the number of events in queue 5 at an instant in time (visible via the NUMVAL bit in QSTAT5) equals or exceeds the value specified by Q5. The default is 16 (maximum allowed). Disables the threshold errors. Reserved |
| 7-5 | Reserved | 0 | Reserved |
| 4-0 | Q4 | 0-10h 11h 12h-1Fh | Queue threshold for queue 4 value. The QTHRCD0 bit in the EDMA3CC error register (CCERR) and the THRXCD bit in the queue status register 4 (QSTAT4) are set when the number of events in queue 4 at an instant in time (visible via the NUMVAL bit in QSTAT4) equals or exceeds the value specified by Q4. The default is 16 (maximum allowed). Disables the threshold errors. Reserved |

4.2.4.5 EDMA3CC Status Register (CCSTAT)

The EDMA3CC status register (CCSTAT) has a number of status bits that reflect which parts of the EDMA3CC logic is active at any given instant of time. The CCSTAT is shown in [Figure 4-25](#) and described in [Table 4-26](#).

Figure 4-25. EDMA3CC Status Register (CCSTAT)



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

Table 4-26. EDMA3CC Status Register (CCSTAT) Field Descriptions

| Bit | Field | Value | Description |
|-------|----------|-------|---|
| 31-22 | Reserved | 0 | Reserved |
| 21 | QUEACTV5 | 0 | Queue 5 active. No events are queued in queue 5. |
| | | 1 | At least one TR is queued in queue 5. |
| 20 | QUEACTV4 | 0 | Queue 4 active. No events are queued in queue 4. |
| | | 1i HA | At least one TR is queued in queue 4. |
| 19 | QUEACTV3 | 0 | Queue 3 active. No events are queued in queue 3. |
| | | 1 | At least one TR is queued in queue 3. |
| 18 | QUEACTV2 | 0 | Queue 2 active. No events are queued in queue 2. |
| | | 1 | At least one TR is queued in queue 2. |
| 17 | QUEACTV1 | 0 | Queue 1 active. No events are queued in queue 1. |
| | | 1 | At least one TR is queued in queue 1. |
| 16 | QUEACTV0 | 0 | Queue 0 active. No events are queued in queue 0. |
| | | 1 | At least one TR is queued in queue 0. |
| 15-14 | Reserved | 0 | Reserved |

Table 4-26. EDMA3CC Status Register (CCSTAT) Field Descriptions (continued)

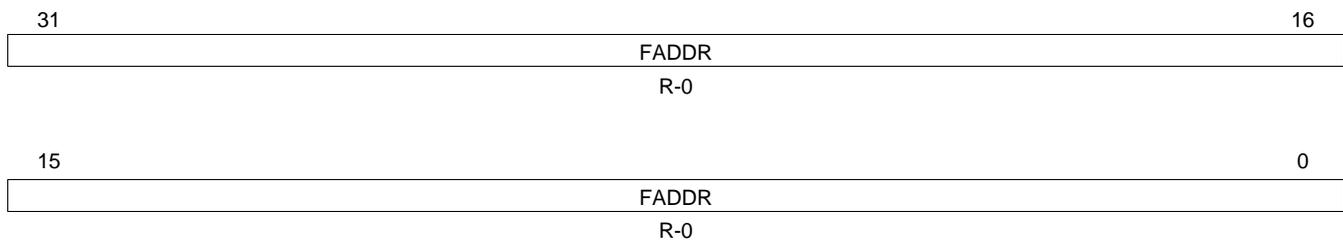
| Bit | Field | Value | Description |
|------|-----------|--------|--|
| 13-8 | COMPACTV | 0-3Fh | Completion request active. The COMPACTV field reflects the count for the number of completion requests submitted to the transfer controllers. This count increments every time a TR is submitted and is programmed to report completion (the TCINTEN or TCCCHEN bits in OPT in the parameter entry associated with the TR are set). The counter decrements for every valid TCC received back from the transfer controllers. If at any time the count reaches a value of 63, the EDMA3CC does not service any new TRs until the count is less than 63 (or return a transfer completion code from a transfer controller, which would decrement the count). |
| | | 0 | No completion requests outstanding. |
| | | 1h-3Fh | Total of 1 completion request to 63 completion requests are outstanding. |
| 7-5 | Reserved | 0 | Reserved |
| 4 | ACTV | | Channel controller active. Channel controller active is a logical-OR of each of the *ACTV bits. The ACTV bit must remain high through the life of a TR. |
| | | 0 | Channel is idle. |
| | | 1 | Channel is busy. |
| 3 | WSTATACTV | | Write status interface active. |
| | | 0 | Write status req is idle and write status FIFO is idle. |
| | | 1 | Either the write status request is active or additional write status responses are pending in the write status FIFO. |
| 2 | TRACTV | | Transfer request active. |
| | | 0 | Transfer request processing/submission logic is inactive. |
| | | 1 | Transfer request processing/submission logic is active. |
| 1 | QEVTACTV | | QDMA event active. |
| | | 0 | No enabled QDMA events are active within the EDMA3CC. |
| | | 1 | At least one enabled QDMA event (QER) is active within the EDMA3CC. |
| 0 | EVTACTV | | DMA event active. |
| | | 0 | No enabled DMA events are active within the EDMA3CC. |
| | | 1 | At least one enabled DMA event (ER and EER, ESR, CER) is active within the EDMA3CC. |

4.2.5 Memory Protection Address Space

4.2.5.1 Memory Protection Fault Address Register (MPFAR)

The memory protection fault address register (MPFAR) is shown in [Figure 4-26](#) and described in [Table 4-27](#). A CPU write of 1 to the MPFCLR bit in the memory protection fault command register (MPFCR) causes any error conditions stored in MPFAR to be cleared.

Figure 4-26. Memory Protection Fault Address Register (MPFAR)



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

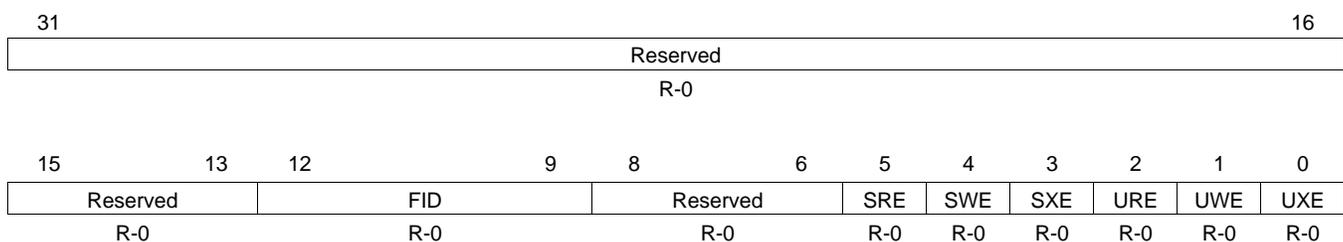
Table 4-27. Memory Protection Fault Address Register (MPFAR) Field Descriptions

| Bit | Field | Value | Description |
|------|-------|--------------|---|
| 31-0 | FADDR | 0-FFFF FFFFh | Fault address. This 32-bit read-only status register contains the fault address when a memory protection violation is detected. This register can only be cleared via the memory protection fault command register (MPFCR). |

4.2.5.2 Memory Protection Fault Status Register (MPFSR)

The memory protection fault status register (MPFSR) is shown in [Figure 4-27](#) and described in [Table 4-28](#). A CPU write of 1 to the MPFCLR bit in the memory protection fault command register (MPFCR) causes any error conditions stored in MPFSR to be cleared.

Figure 4-27. Memory Protection Fault Status Register (MPFSR)



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

Table 4-28. Memory Protection Fault Status Register (MPFSR) Field Descriptions

| Bit | Field | Value | Description |
|-------|----------|--------|--|
| 31-13 | Reserved | 0 | Reserved |
| 12-9 | FID | 0-Fh | Faulted identification. FID contains valid information if any of the MP error bits (UXE, UWE, URE, SXE, SWE, SRE) are nonzero (that is, if an error has been detected.) The FID field contains the privilege ID for the specific request/requestor that resulted in an MP error. |
| 8-6 | Reserved | 0 | Reserved |
| 5 | SRE | 0 1 | Supervisor read error. No error detected. Supervisor level task attempted to read from a MP page without SR permissions. |

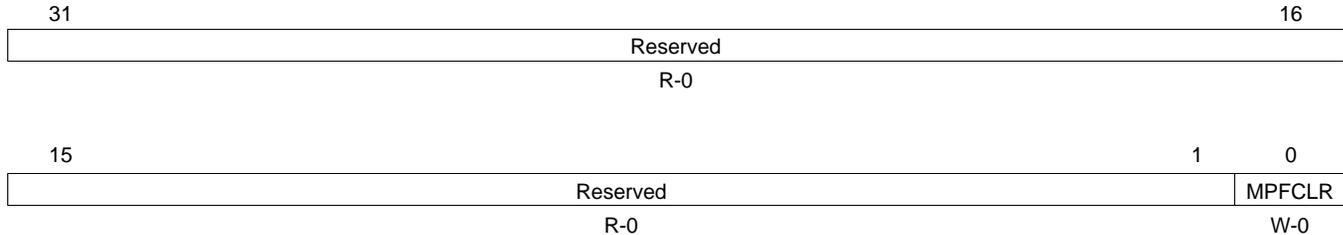
Table 4-28. Memory Protection Fault Status Register (MPFSR) Field Descriptions (continued)

| Bit | Field | Value | Description |
|-----|-------|-------|---|
| 4 | SWE | 0 | Supervisor write error. No error detected. |
| | | 1 | Supervisor level task attempted to write to a MP page without SW permissions. |
| 3 | SXE | 0 | Supervisor execute error. No error detected. |
| | | 1 | Supervisor level task attempted to execute from a MP page without SX permissions. |
| 2 | URE | 0 | User read error. No error detected. |
| | | 1 | User level task attempted to read from a MP page without UR permissions. |
| 1 | UWE | 0 | User write error. No error detected. |
| | | 1 | User level task attempted to write to a MP page without UW permissions. |
| 0 | UXE | 0 | User execute error. No error detected. |
| | | 1 | User level task attempted to execute from a MP page without UX permissions. |

4.2.5.3 Memory Protection Fault Command Register (MPFCR)

The memory protection fault command register (MPFCR) is shown in [Figure 4-28](#) and described in [Table 4-29](#).

Figure 4-28. Memory Protection Fault Command Register (MPFCR)



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

Table 4-29. Memory Protection Fault Command Register (MPFCR) Field Descriptions

| Bit | Field | Value | Description |
|------|----------|-------|--|
| 31-1 | Reserved | 0 | Reserved |
| 0 | MPFCLR | 0 | Fault clear register. CPU write of 0 has no effect. |
| | | 1 | CPU write of 1 to the MPFCLR bit causes any error conditions stored in the memory protection fault address register (MPFAR) and the memory protection fault status register (MPFSR) to be cleared. |

4.2.6 DMA Channel Registers

The following sets of registers pertain to the 64 DMA channels. The 64 DMA channels consist of a set of registers (with exception of DMAQNUM n) that each have 64 bits and the bit position of each register matches the DMA channel number. Each register is named with the format *reg_name* that corresponds to DMA channels 0 through 31 and *reg_name_High* that corresponds to DMA channels 32 through 64.

For example, the event register (ER) corresponds to DMA channel 0 through 31 and the event register high register (ERH) corresponds to DMA channel 32 through 63. The register is typically called the event register.

The DMA channel registers are accessible via read/writes to the global address range. They are also accessible via read/writes to the shadow address range. The read/write ability to the registers in the shadow region are controlled by the DMA region access registers (DRAEm/DRAEHm). The registers are described in [Section 4.2.3.1](#) and the details for shadow region/global region usage is explained in [Section 2.7](#).

4.2.6.1 Event Registers (ER, ERH)

All external events are captured in the event register (ER/ERH). The events are latched even when the events are not enabled. If the event bit corresponding to the latched event is enabled (EER.En/EERH.En = 1), then the event is evaluated by the EDMA3CC logic for an associated transfer request submission to the transfer controllers. The event register bits are automatically cleared (ER.En/ERH.En = 0) once the corresponding events are prioritized and serviced. If ER.En/ERH.En are already set and another event is received on the same channel/event, then the corresponding event is latched in the event miss register (EMR.En/EMRH.En), provided that the event was enabled (EER.En/EERH.En = 1).

Event n can be cleared by the CPU writing a 1 to corresponding event bit in the event clear register (ECR/ECRH). The setting of an event is a higher priority relative to clear operations (via hardware or software). If set and clear conditions occur concurrently, the set condition wins. If the event was previously set, then EMR/EMRH would be set because an event is lost. If the event was previously clear, then the event remains set and is prioritized for submission to the event queues.

Table A 1 provides the type of synchronization events and the EDMA3CC channels associated to each of these external events.

The ER is shown in [Figure 4-30](#) and described in [Table 4-31](#). The ERH is shown in [Figure 4-31](#) and described in [Table 4-32](#).

Figure 4-30. Event Register (ER)

| | | | | | | | | | | | | | | | |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| E31 | E30 | E29 | E28 | E27 | E26 | E25 | E24 | E23 | E22 | E21 | E20 | 19 | E18 | E17 | E16 |
| R-0 |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| E15 | E14 | E13 | E12 | E11 | E10 | E9 | E8 | E7 | E6 | E5 | E4 | E3 | E2 | E1 | E0 |
| R-0 |

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

Table 4-31. Event Register (ER) Field Descriptions

| Bit | Field | Value | Description |
|------|-------|-------|---|
| 31-0 | E_n | 0 | Event 0-31. Events 0-31 are captured by the EDMA3CC and are latched into ER. The events are set ($E_n = 1$) even when events are disabled ($E_n = 0$ in the event enable register, EER). EDMA3CC event is not asserted. |
| | | 1 | EDMA3CC event is asserted. Corresponding DMA event is prioritized versus other pending DMA/QDMA events for submission to the EDMA3TC. |

Figure 4-31. Event Register High (ERH)

| | | | | | | | | | | | | | | | |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| E63 | E62 | E61 | E60 | E59 | E58 | E57 | E56 | E55 | E54 | E53 | E52 | E51 | E50 | E49 | E48 |
| R-0 |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| E47 | E46 | E45 | E44 | E43 | E42 | E41 | E40 | E39 | E38 | E37 | E36 | E35 | E34 | E33 | E32 |
| R-0 |

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

Table 4-32. Event Register High (ERH) Field Descriptions

| Bit | Field | Value | Description |
|------|-------|-------|--|
| 31-0 | E_n | 0 | Event 32-63. Events 32-63 are captured by the EDMA3CC and are latched into ERH. The events are set ($E_n = 1$) even when events are disabled ($E_n = 0$ in the event enable register high, EERH). EDMA3CC event is not asserted. |
| | | 1 | EDMA3CC event is asserted. Corresponding DMA event is prioritized versus other pending DMA/QDMA events for submission to the EDMA3TC. |

4.2.6.2 Event Clear Registers (ECR, ECRH)

Once an event has been posted in the event registers (ER/ERH), the event is cleared in two ways. If the event is enabled in the event enable register (EER/EERH) and the EDMA3CC submits a transfer request for the event to the EDMA3TC, it clears the corresponding event bit in the event register. If the event is disabled in the event enable register (EER/EERH), the CPU can clear the event by way of the event clear registers (ECR/ECRH).

Writing a 1 to any of the bits clears the corresponding event; writing a 0 has no effect. Once an event bit is set in the event register, it remains set until EDMA3CC submits a transfer request for that event or the CPU clears the event by setting the corresponding bit in ECR/ECRH.

The ECR is shown in [Figure 4-32](#) and described in [Table 4-33](#). The ECRH is shown in [Figure 4-33](#) and described in [Table 4-34](#).

Figure 4-32. Event Clear Register (ECR)

| | | | | | | | | | | | | | | | |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| E31 | E30 | E29 | E28 | E27 | E26 | E25 | E24 | E23 | E22 | E21 | E20 | E19 | E18 | E17 | E16 |
| W-0 |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| E15 | E14 | E13 | E12 | E11 | E10 | E9 | E8 | E7 | E6 | E5 | E4 | E3 | E2 | E1 | E0 |
| W-0 |

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

Table 4-33. Event Clear Register (ECR) Field Descriptions

| Bit | Field | Value | Description |
|------|-----------|-------|--|
| 31-0 | <i>En</i> | | Event clear for event 0-31. Any of the event bits in ECR is set to clear the event (<i>En</i>) in the event register (ER). A write of 0 has no effect. |
| | | 0 | No effect. |
| | | 1 | EDMA3CC event is cleared in the event register (ER). |

Figure 4-33. Event Clear Register High (ECRH)

| | | | | | | | | | | | | | | | |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| E63 | E62 | E61 | E60 | E59 | E58 | E57 | E56 | E55 | E54 | E53 | E52 | E51 | E50 | E49 | E48 |
| W-0 |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| E47 | E46 | E45 | E44 | E43 | E42 | E41 | E40 | E39 | E38 | E37 | E36 | E35 | E34 | E33 | E32 |
| W-0 |

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

Table 4-34. Event Clear Register High (ECRH) Field Descriptions

| Bit | Field | Value | Description |
|------|-----------|-------|---|
| 31-0 | <i>En</i> | | Event clear for event 32-63. Any of the event bits in ECRH are set to clear the event (<i>En</i>) in the event register high (ERH). A write of 0 has no effect. |
| | | 0 | No effect. |
| | | 1 | EDMA3CC event is cleared in the event register high (ERH). |

4.2.6.3 Event Set Registers (ESR, ESRH)

The event set registers (ESR/ESRH) allow the CPU (EDMA3 programmers) to manually set events to initiate DMA transfer requests. CPU writes of 1 to any event set register (E_n) bits set the corresponding bits in the registers. The set event is evaluated by the EDMA3CC logic for an associated transfer request submission to the transfer controllers. Writing a 0 has no effect.

The event set registers operate independent of the event registers (ER/ERH), and a write of 1 is always considered a valid event regardless of whether the event is enabled (the corresponding event bits are set or cleared in EER. E_n /EERH. E_n).

Once the event is set in the event set registers, it cannot be cleared by CPU writes, in other words, the event clear registers (ECR/ECRH) have no effect on the state of ESR/ESRH. The bits are cleared only once the transfer request corresponding to the event has been submitted to the transfer controller. The setting of an event is a higher priority relative to clear operations (via hardware). If set and clear conditions occur concurrently, the set condition wins. If the event was previously set, then EMR/EMRH would be set because an event is lost. If the event was previously clear, then the event remains set and is prioritized for submission to the event queues.

Manually-triggered transfers via writes to ESR/ESRH allow the CPU to submit DMA requests in the system, these are relevant for memory-to-memory transfer scenarios. If the ESR. E_n /ESRH. E_n bit is already set and another CPU write of 1 is attempted to the same bit, then the corresponding event is latched in the event missed registers (EMR. E_n /EMRH. $E_n = 1$).

The ESR is shown in [Figure 4-34](#) and described in [Table 4-35](#). The ESRH is shown in [Figure 4-35](#) and described in [Table 4-36](#).

Figure 4-34. Event Set Register (ESR)

| | | | | | | | | | | | | | | | |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| E31 | E30 | E29 | E28 | E27 | E26 | E25 | E24 | E23 | E22 | E21 | E20 | E19 | E18 | E17 | E16 |
| RW-0 |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| E15 | E14 | E13 | E12 | E11 | E10 | E9 | E8 | E7 | E6 | E5 | E4 | E3 | E2 | E1 | E0 |
| RW-0 |

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

Table 4-35. Event Set Register (ESR) Field Descriptions

| Bit | Field | Value | Description |
|------|-------|-------|--|
| 31-0 | E_n | 0 | Event set for event 0-31. No effect. |
| | | 1 | Corresponding DMA event is prioritized versus other pending DMA/QDMA events for submission to the EDMA3TC. |

Figure 4-35. Event Set Register High (ESRH)

| | | | | | | | | | | | | | | | |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| E63 | E62 | E61 | E60 | E59 | E58 | E57 | E56 | E55 | E54 | E53 | E52 | E51 | E50 | E49 | E48 |
| RW-0 |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| E47 | E46 | E45 | E44 | E43 | E42 | E41 | E40 | E39 | E38 | E37 | E36 | E35 | E34 | E33 | E32 |
| RW-0 |

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

Table 4-36. Event Set Register High (ESRH) Field Descriptions

| Bit | Field | Value | Description |
|------|-----------|-------|--|
| 31-0 | <i>En</i> | 0 | Event set for event 32-63. No effect . |
| | | 1 | Corresponding DMA event is prioritized versus other pending DMA/QDMA events for submission to the EDMA3TC. |

4.2.6.4 Chained Event Registers (CER, CERH)

When the **OPTIONS** parameter for a PaRAM entry is programmed to returned a chained completion code (**ITCCHEN** = 1 and/or **TCCHEN** = 1), then the value dictated by the **TCC[5:0]** (also programmed in **OPT**) forces the corresponding event bit to be set in the chained event registers (**CER/CERH**). The set chained event is evaluated by the EDMA3CC logic for an associated transfer request submission to the transfer controllers. This results in a chained-triggered transfer.

The chained event registers do not have any enables. The generation of a chained event is essentially enabled by the PaRAM entry that has been configured for intermediate and/or final chaining on transfer completion. The *En* bit is set (regardless of the state of **EER.En/EERH.En**) when a chained completion code is returned from one of the transfer controllers or is generated by the EDMA3CC via the early completion path. The bits in the chained event register are cleared when the corresponding events are prioritized and serviced.

If the *En* bit is already set and another chaining completion code is return for the same event, then the corresponding event is latched in the event missed registers (**EMR.En/EMRH.En** = 1). The setting of an event is a higher priority relative to clear operations (via hardware). If set and clear conditions occur concurrently, the set condition wins. If the event was previously set, then **EMR/EMRH** would be set because an event is lost. If the event was previously clear, then the event remains set and is prioritized for submission to the event queues.

The **CER** is shown in [Figure 4-36](#) and described in [Table 4-37](#). The **CERH** is shown in [Figure 4-37](#) and described in [Table 4-38](#).

Figure 4-36. Chained Event Register (CER)

| | | | | | | | | | | | | | | | |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| E31 | E30 | E29 | E28 | E27 | E26 | E25 | E24 | E23 | E22 | E21 | E20 | E19 | E18 | E17 | E16 |
| R-0 |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| E15 | E14 | E13 | E12 | E11 | E10 | E9 | E8 | E7 | E6 | E5 | E4 | E3 | E2 | E1 | E0 |
| R-0 |

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

Table 4-37. Chained Event Register (CER) Field Descriptions

| Bit | Field | Value | Description |
|------|-------|-------|--|
| 31-0 | E_n | 0 | Chained event for event 0-31. No effect. |
| | | 1 | Corresponding DMA event is prioritized versus other pending DMA/QDMA events for submission to the EDMA3TC. |

Figure 4-37. Chained Event Register High (CERH)

| | | | | | | | | | | | | | | | |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| E63 | E62 | E61 | E60 | E59 | E58 | E57 | E56 | E55 | E54 | E53 | E52 | E51 | E50 | E49 | E48 |
| R-0 |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| E47 | E46 | E45 | E44 | E43 | E42 | E41 | E40 | E39 | E38 | E37 | E36 | E35 | E34 | E33 | E32 |
| R-0 |

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

Table 4-38. Chained Event Register High (CERH) Field Descriptions

| Bit | Field | Value | Description |
|------|-------|-------|--|
| 31-0 | E_n | 0 | Chained event set for event 32-63. No effect. |
| | | 1 | Corresponding DMA event is prioritized versus other pending DMA/QDMA events for submission to the EDMA3TC. |

4.2.6.5 Event Enable Registers (EER, EERH)

The EDMA3CC provides the option of selectively enabling/disabling each event in the event registers (ER/ERH) by using the event enable registers (EER/EERH). If an event bit in EER/EERH is set (using the event enable set registers, EESR/EESRH), it enables that corresponding event. Alternatively, if an event bit in EER/EERH is cleared (using the event enable clear registers, EECR/EECRH), it disables the corresponding event.

The event registers latch all events that are captured by EDMA3CC, even if the events are disabled (although EDMA3CC does not process it). Enabling an event with a pending event already set in the event registers enables the EDMA3CC to process the already set event like any other new event. The EER/EERH settings do not have any effect on chained events ($CER.En/CERH.En = 1$) and manually set events ($ESR.En/ESRH.En = 1$).

The EER is shown in [Figure 4-38](#) and described in [Table 4-39](#). The EERH is shown in [Figure 4-39](#) and described in [Table 4-40](#).

Figure 4-38. Event Enable Register (EER)

| | | | | | | | | | | | | | | | |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| E31 | E30 | E29 | E28 | E27 | E26 | E25 | E24 | E23 | E22 | E21 | E20 | E19 | E18 | E17 | E16 |
| R-0 |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| E15 | E14 | E13 | E12 | E11 | E10 | E9 | E8 | E7 | E6 | E5 | E4 | E3 | E2 | E1 | E0 |
| R-0 |

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

Table 4-39. Event Enable Register (EER) Field Descriptions

| Bit | Field | Value | Description |
|------|-------|-------|--|
| 31-0 | En | 0 | Event enable for events 0-31. Event is not enabled. An external event latched in the event register (ER) is not evaluated by the EDMA3CC. |
| | | 1 | Event is enabled. An external event latched in the event register (ER) is evaluated by the EDMA3CC. |

Figure 4-39. Event Enable Register High (EERH)

| | | | | | | | | | | | | | | | |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| E63 | E62 | E61 | E60 | E59 | E58 | E57 | E56 | E55 | E54 | E53 | E52 | E51 | E50 | E49 | E48 |
| R-0 |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| E47 | E46 | E45 | E44 | E43 | E42 | E41 | E40 | E39 | E38 | E37 | E36 | E35 | E34 | E33 | E32 |
| R-0 |

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

Table 4-40. Event Enable Register High (EERH) Field Descriptions

| Bit | Field | Value | Description |
|------|-------|-------|---|
| 31-0 | En | 0 | Event enable for events 32-63. Event is not enabled. An external event latched in the event register high (ERH) is not evaluated by the EDMA3CC. |
| | | 1 | Event is enabled. An external event latched in the event register high (ERH) is evaluated by the EDMA3CC. |

4.2.6.6 Event Enable Clear Register (EECR, EECRH)

The event enable registers (EER/EERH) cannot be modified by directly writing to them. The intent is to ease the software burden for the case where multiple tasks are attempting to simultaneously modify these registers. The event enable clear registers (EECR/EECRH) are used to disable events. Writes of 1 to the bits in EECR/EECRH clear the corresponding event bits in EER/EERH; writes of 0 have no effect.

The EECR is shown in [Figure 4-40](#) and described in [Table 4-41](#). The EECRH is shown in [Figure 4-41](#) and described in [Table 4-42](#).

Figure 4-40. Event Enable Clear Register (EECR)

| | | | | | | | | | | | | | | | |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| E31 | E30 | E29 | E28 | E27 | E26 | E25 | E24 | E23 | E22 | E21 | E20 | E19 | E18 | E17 | E16 |
| W-0 |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| E15 | E14 | E13 | E12 | E11 | E10 | E9 | E8 | E7 | E6 | E5 | E4 | E3 | E2 | E1 | E0 |
| W-0 |

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

Table 4-41. Event Enable Clear Register (EECR) Field Descriptions

| Bit | Field | Value | Description |
|------|-------|-------|---|
| 31-0 | E_n | 0 | Event enable clear for events 0-31. No effect. |
| | | 1 | Event is disabled. Corresponding bit in the event enable register (EER) is cleared ($E_n = 0$). |

Figure 4-41. Event Enable Clear Register High (EECRH)

| | | | | | | | | | | | | | | | |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| E63 | E62 | E61 | E60 | E59 | E58 | E57 | E56 | E55 | E54 | E53 | E52 | E51 | E50 | E49 | E48 |
| W-0 |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| E47 | E46 | E45 | E44 | E43 | E42 | E41 | E40 | E39 | E38 | E37 | E36 | E35 | E34 | E33 | E32 |
| W-0 |

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

Table 4-42. Event Enable Clear Register High (EECRH) Field Descriptions

| Bit | Field | Value | Description |
|------|-------|-------|---|
| 31-0 | E_n | 0 | Event enable clear for events 32-63. No effect. |
| | | 1 | Event is disabled. Corresponding bit in the event enable register high (EERH) is cleared ($E_n = 0$). |

4.2.6.7 Event Enable Set Registers (EESR, EESRH)

The event enable registers (EER/EERH) cannot be modified by directly writing to them. The intent is to ease the software burden for the case where multiple tasks are attempting to simultaneously modify these registers. The event enable set registers (EESR/EESRH) are used to enable events. Writes of 1 to the bits in EESR/EESRH set the corresponding event bits in EER/EERH; writes of 0 have no effect.

The EESR is shown in Figure 4-42 and described in Table 4-43. The EESRH is shown in Figure 4-43 and described in Table 4-44.

Figure 4-42. Event Enable Set Register (EESR)

| | | | | | | | | | | | | | | | |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| E31 | E30 | E29 | E28 | E27 | E26 | E25 | E24 | E23 | E22 | E21 | E20 | E19 | E18 | E17 | E16 |
| W-0 |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| E15 | E14 | E13 | E12 | E11 | E10 | E9 | E8 | E7 | E6 | E5 | E4 | E3 | E2 | E1 | E0 |
| W-0 |

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

Table 4-43. Event Enable Set Register (EESR) Field Descriptions

| Bit | Field | Value | Description |
|------|-------|-------|--|
| 31-0 | E_n | 0 | Event enable set for events 0-31. No effect. |
| | | 1 | Event is enabled. Corresponding bit in the event enable register (EER) is set ($E_n = 1$). |

Figure 4-43. Event Enable Set Register High (EESRH)

| | | | | | | | | | | | | | | | |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| E63 | E62 | E61 | E60 | E59 | E58 | E57 | E56 | E55 | E54 | E53 | E52 | E51 | E50 | E49 | E48 |
| W-0 |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| E47 | E46 | E45 | E44 | E43 | E42 | E41 | E40 | E39 | E38 | E37 | E36 | E35 | E34 | E33 | E32 |
| W-0 |

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

Table 4-44. Event Enable Set Register High (EESRH) Field Descriptions

| Bit | Field | Value | Description |
|------|-------|-------|--|
| 31-0 | E_n | 0 | Event enable set for events 32-63. No effect. |
| | | 1 | Event is enabled. Corresponding bit in the event enable register high (EERH) is set ($E_n = 1$). |

4.2.6.8 Secondary Event Registers (SER, SERH)

The secondary event registers (SER/SERH) provide information on the state of a DMA channel or event (0 through 63). If the EDMA3CC receives a TR synchronization due to a manual-trigger, event-trigger, or chained-trigger source ($ESR.En/ESRH.En = 1$, $ER.En/ERH.En = 1$, or $CER.En/CERH.En = 1$), which results in the setting of a corresponding event bit in SER/SERH ($SER.En/SERH.En = 1$), it implies that the corresponding DMA event is in the queue.

Once a bit corresponding to an event is set in SER/SERH, the EDMA3CC does not prioritize additional events on the same DMA channel. Depending on the condition that lead to the setting of the SER bits, either the EDMA3CC hardware or the software (using SECR/SECRH) needs to clear the SER/SERH bits for the EDMA3CC to evaluate subsequent events (subsequent transfers) on the same channel. For additional conditions that can cause the secondary event registers to be set, see [Section 1.1](#).

The SER is shown in [Figure 4-44](#) and described in [Table 4-45](#). The SERH is shown in [Figure 4-45](#) and described in [Table 4-46](#).

Figure 4-44. Secondary Event Register (SER)

| | | | | | | | | | | | | | | | |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| E31 | E30 | E29 | E28 | E27 | E26 | E25 | E24 | E23 | E22 | E21 | E20 | E19 | E18 | E17 | E16 |
| R-0 |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| E15 | E14 | E13 | E12 | E11 | E10 | E9 | E8 | E7 | E6 | E5 | E4 | E3 | E2 | E1 | E0 |
| R-0 |

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

Table 4-45. Secondary Event Register (SER) Field Descriptions

| Bit | Field | Value | Description |
|------|-------|-------|--|
| 31-0 | En | | Secondary event register. The secondary event register is used along with the event register (ER) to provide information on the state of an event. |
| | | 0 | Event is not currently stored in the event queue. |
| | | 1 | Event is currently stored in the event queue. Event arbiter does not prioritize additional events. |

Figure 4-45. Secondary Event Register High (SERH)

| | | | | | | | | | | | | | | | |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| E63 | E62 | E61 | E60 | E59 | E58 | E57 | E56 | E55 | E54 | E53 | E52 | E51 | E50 | E49 | E48 |
| R-0 |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| E47 | E46 | E45 | E44 | E43 | E42 | E41 | E40 | E39 | E38 | E37 | E36 | E35 | E34 | E33 | E32 |
| R-0 |

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

Table 4-46. Secondary Event Register High (SERH) Field Descriptions

| Bit | Field | Value | Description |
|------|-------|-------|--|
| 31-0 | En | | Secondary event register. The secondary event register is used along with the event register high (ERH) to provide information on the state of an event. |
| | | 0 | Event is not currently stored in the event queue. |
| | | 1 | Event is currently stored in the event queue. Event submission/prioritization logic does not prioritize additional events. |

4.2.6.9 Secondary Event Clear Registers (SECR, SECRH)

The secondary event clear registers (SECR/SECRH) clear the status of the secondary event registers (SER/SERH). CPU writes of 1 clear the corresponding set bits in SER/SERH. Writes of 0 have no effect.

The SECR is shown in [Figure 4-46](#) and described in [Table 4-47](#). The SECRH is shown in [Figure 4-47](#) and described in [Table 4-48](#).

Figure 4-46. Secondary Event Clear Register (SECR)

| | | | | | | | | | | | | | | | |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| E31 | E30 | E29 | E28 | E27 | E26 | E25 | E24 | E23 | E22 | E21 | E20 | E19 | E18 | E17 | E16 |
| W-0 |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| E15 | E14 | E13 | E12 | E11 | E10 | E9 | E8 | E7 | E6 | E5 | E4 | E3 | E2 | E1 | E0 |
| W-0 |

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

Table 4-47. Secondary Event Clear Register (SECR) Field Descriptions

| Bit | Field | Value | Description |
|------|-------|-------|---|
| 31-0 | E_n | 0 | Secondary event clear register No effect. |
| | | 1 | Corresponding bit in the secondary event register (SER) is cleared ($E_n = 0$). |

Figure 4-47. Secondary Event Clear Register High (SECRH)

| | | | | | | | | | | | | | | | |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| E63 | E62 | E61 | E60 | E59 | E58 | E57 | E56 | E55 | E54 | E53 | E52 | E51 | E50 | E49 | E48 |
| W-0 |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| E47 | E46 | E45 | E44 | E43 | E42 | E41 | E40 | E39 | E38 | E37 | E36 | E35 | E34 | E33 | E32 |
| W-0 |

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

Table 4-48. Secondary Event Clear Register High (SECRH) Field Descriptions

| Bit | Field | Value | Description |
|------|-------|-------|--|
| 31-0 | E_n | 0 | Secondary event clear register. No effect. |
| | | 1 | Corresponding bit in the secondary event registers high (SERH) is cleared ($E_n = 0$). |

4.2.7 Interrupt Registers

All DMA/QDMA channels can be set to assert an EDMA3CC completion interrupt to the CPU on transfer completion, by appropriately configuring the PaRAM entry associated with the channels. The following set of registers is used for the transfer completion interrupt reporting/generating by the EDMA3CC. For more details on EDMA3CC completion interrupt generation, see [Section 2.9](#).

4.2.7.1 Interrupt Enable Registers (IER, IERH)

Interrupt enable registers (IER/IERH) are used to enable/disable the transfer completion interrupt generation by the EDMA3CC for all DMA/QDMA channels. The IER/IERH cannot be written to directly. To set any interrupt bit in IER/IERH, a 1 must be written to the corresponding interrupt bit in the interrupt enable set registers (IESR/IESRH). Similarly, to clear any interrupt bit in IER/IERH, a 1 must be written to the corresponding interrupt bit in the interrupt enable clear registers (IECR/IECRH).

The IER is shown in [Figure 4-48](#) and described in [Table 4-49](#). The IERH is shown in [Figure 4-49](#) and described in [Table 4-50](#).

Figure 4-48. Interrupt Enable Register (IER)

| | | | | | | | | | | | | | | | |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| I31 | I30 | I29 | I28 | I27 | I26 | I25 | I24 | I23 | I22 | I21 | I20 | I19 | I18 | I17 | I16 |
| R-0 |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| I15 | I14 | I13 | I12 | I11 | I10 | I9 | I8 | I7 | I6 | I5 | I4 | I3 | I2 | I1 | I0 |
| R-0 |

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

Table 4-49. Interrupt Enable Register (IER) Field Descriptions

| Bit | Field | Value | Description |
|------|-----------|-------|--|
| 31-0 | <i>En</i> | 0 | Interrupt enable for channels 0-31. Interrupt is not enabled. |
| | | 1 | Interrupt is enabled. |

Figure 4-49. Interrupt Enable Register High (IERH)

| | | | | | | | | | | | | | | | |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| I63 | I62 | I61 | I60 | I59 | I58 | I57 | I56 | I55 | I54 | I53 | I52 | I51 | I50 | I49 | I48 |
| R-0 |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| I47 | I46 | I45 | I44 | I43 | I42 | I41 | I40 | I39 | I38 | I37 | I36 | I35 | I34 | I33 | I32 |
| R-0 |

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

Table 4-50. Interrupt Enable Register High (IERH) Field Descriptions

| Bit | Field | Value | Description |
|------|-----------|-------|---|
| 31-0 | <i>En</i> | 0 | Interrupt enable for channels 32-63. Interrupt is not enabled. |
| | | 1 | Interrupt is enabled. |

4.2.7.2 Interrupt Enable Clear Register (IECR, IECRH)

The interrupt enable clear registers (IECR/IECRH) are used to clear interrupts. Writes of 1 to the bits in IECR/IECRH clear the corresponding interrupt bits in the interrupt enable registers (IER/IERH); writes of 0 have no effect.

The IECR is shown in [Figure 4-50](#) and described in [Table 4-51](#). The IECRH is shown in [Figure 4-51](#) and described in [Table 4-52](#).

Figure 4-50. Interrupt Enable Clear Register (IECR)

| | | | | | | | | | | | | | | | |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| I31 | I30 | I29 | I28 | I27 | I26 | I25 | I24 | I23 | I22 | I21 | I20 | I19 | I18 | I17 | I16 |
| W-0 |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| I15 | I14 | I13 | I12 | I11 | I10 | I9 | I8 | I7 | I6 | I5 | I4 | I3 | I2 | I1 | I0 |
| W-0 |

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

Table 4-51. Interrupt Enable Clear Register (IECR) Field Descriptions

| Bit | Field | Value | Description |
|------|-----------|-------|---|
| 31-0 | <i>En</i> | 0 | Interrupt enable clear for channels 0-31. No effect |
| | | 1 | Corresponding bit in the interrupt enable register (IER) is cleared (<i>In</i> = 0). |

Figure 4-51. Interrupt Enable Clear Register High (IECRH)

| | | | | | | | | | | | | | | | |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| I63 | I62 | I61 | I60 | I59 | I58 | I57 | I56 | I55 | I54 | I53 | I52 | I51 | I50 | I49 | I48 |
| W-0 |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| I47 | I46 | I45 | I44 | I43 | I42 | I41 | I40 | I39 | I38 | I37 | I36 | I35 | I34 | I33 | I32 |
| W-0 |

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

Table 4-52. Interrupt Enable Clear Register High (IECRH) Field Descriptions

| Bit | Field | Value | Description |
|------|-----------|-------|---|
| 31-0 | <i>En</i> | 0 | Interrupt enable clear for channels 32-63. No effect. |
| | | 1 | Corresponding bit in the interrupt enable register high (IERH) is cleared (<i>In</i> = 0). |

4.2.7.3 Interrupt Enable Set Registers (IESR, IESRH)

The interrupt enable set registers (IESR/IESRH) are used to enable interrupts. Writes of 1 to the bits in IESR/IESRH set the corresponding interrupt bits in the interrupt enable registers (IER/IERH); writes of 0 have no effect.

The IESR is shown in Figure 4-52 and described in Table 4-53. The IESRH is shown in Figure 4-53 and described in Table 4-54.

Figure 4-52. Interrupt Enable Set Register (IESR)

| | | | | | | | | | | | | | | | |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| I31 | I30 | I29 | I28 | I27 | I26 | I25 | I24 | I23 | I22 | I21 | I20 | I19 | I18 | I17 | I16 |
| W-0 |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| I15 | I14 | I13 | I12 | I11 | I10 | I9 | I8 | I7 | I6 | I5 | I4 | I3 | I2 | I1 | I0 |
| W-0 |

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

Table 4-53. Interrupt Enable Set Register (IESR) Field Descriptions

| Bit | Field | Value | Description |
|------|-----------|-------|--|
| 31-0 | <i>En</i> | 0 | Interrupt enable set for channels 0-31. No effect. |
| | | 1 | Corresponding bit in the interrupt enable register (IER) is set ($I_n = 1$). |

Figure 4-53. Interrupt Enable Set Register High (IESRH)

| | | | | | | | | | | | | | | | |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| I63 | I62 | I61 | I60 | I59 | I58 | I57 | I56 | I55 | I54 | I53 | I52 | I51 | I50 | I49 | I48 |
| W-0 |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| I47 | I46 | I45 | I44 | I43 | I42 | I41 | I40 | I39 | I38 | I37 | I36 | I35 | I34 | I33 | I32 |
| W-0 |

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

Table 4-54. Interrupt Enable Set Register High (IESRH) Field Descriptions

| Bit | Field | Value | Description |
|------|-----------|-------|--|
| 31-0 | <i>En</i> | 0 | Interrupt enable clear for channels 32-63. No effect. |
| | | 1 | Corresponding bit in the interrupt enable register high (IERH) is set ($I_n = 1$). |

4.2.7.4 Interrupt Pending Register (IPR, IPRH)

If the TCINTEN and/or ITCINTEN bit in the channel option parameter (OPT) is set in the PaRAM entry associated with the channel (DMA or QDMA), then the EDMA3TC (for normal completion) or the EDMA3CC (for early completion) returns a completion code on transfer or intermediate transfer completion. The value of the returned completion code is equal to the TCC bit in OPT for the PaRAM entry associated with the channel.

When an interrupt transfer completion code with $TCC = n$ is detected by the EDMA3CC, then the corresponding bit is set in the interrupt pending register (IPR. I_n , if $n = 0$ to 31; IPRH. I_n , if $n = 32$ to 63). Note that once a bit is set in the interrupt pending registers, it remains set; it is your responsibility to clear these bits. The bits set in IPR/IPRH are cleared by writing a 1 to the corresponding bits in the interrupt clear registers (ICR/ICRH).

The IPR is shown in Figure 4-54 and described in Table 4-55. The IPRH is shown in Figure 4-55 and described in Table 4-56.

Figure 4-54. Interrupt Pending Register (IPR)

| | | | | | | | | | | | | | | | |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| I31 | I30 | I29 | I28 | I27 | I26 | I25 | I24 | I23 | I22 | I21 | I20 | I19 | I18 | I17 | I16 |
| R-0 |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| I15 | I14 | I13 | I12 | I11 | I10 | I9 | I8 | I7 | I6 | I5 | I4 | I3 | I2 | I1 | I0 |
| R-0 |

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

Table 4-55. Interrupt Pending Register (IPR) Field Descriptions

| Bit | Field | Value | Description |
|------|-------|-------|--|
| 31-0 | I_n | 0 | Interrupt pending for $TCC = 0$ -31. |
| | | 1 | Interrupt transfer completion code is not detected or was cleared. |
| | | 1 | Interrupt transfer completion code is detected ($I_n = 1$, $n = EDMA3TC[5:0]$). |

Figure 4-55. Interrupt Pending Register High (IPRH)

| | | | | | | | | | | | | | | | |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| I63 | I62 | I61 | I60 | I59 | I58 | I57 | I56 | I55 | I54 | I53 | I52 | I51 | I50 | I49 | I48 |
| R-0 |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| I47 | I46 | I45 | I44 | I43 | I42 | I41 | I40 | I39 | I38 | I37 | I36 | I35 | I34 | I33 | I32 |
| R-0 |

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

Table 4-56. Interrupt Pending Register High (IPRH) Field Descriptions

| Bit | Field | Value | Description |
|------|-------|-------|--|
| 31-0 | I_n | 0 | Interrupt pending for $TCC = 32$ -63. |
| | | 1 | Interrupt transfer completion code is not detected or was cleared. |
| | | 1 | Interrupt transfer completion code is detected ($I_n = 1$, $n = EDMA3TC[5:0]$). |

4.2.7.5 Interrupt Clear Registers (ICR, ICRH)

The bits in the interrupt pending registers (IPR/IPRH) are cleared by writing a 1 to the corresponding bits in the interrupt clear registers (ICR/ICRH). Writes of 0 have no effect. All set bits in IPR/IPRH must be cleared to allow EDMA3CC to assert additional transfer completion interrupts.

The ICR is shown in [Figure 4-56](#) and described in [Table 4-57](#). The ICRH is shown in [Figure 4-57](#) and described in [Table 4-58](#).

Figure 4-56. Interrupt Clear Register (ICR)

| | | | | | | | | | | | | | | | |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| I31 | I30 | I29 | I28 | I27 | I26 | I25 | I24 | I23 | I22 | I21 | I20 | I19 | I18 | I17 | I16 |
| W-0 |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| I15 | I14 | I13 | I12 | I11 | I10 | I9 | I8 | I7 | I6 | I5 | I4 | I3 | I2 | I1 | I0 |
| W-0 |

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

Table 4-57. Interrupt Clear Register (ICR) Field Descriptions

| Bit | Field | Value | Description |
|------|-----------|-------|--|
| 31-0 | <i>In</i> | 0 | Interrupt clear register for TCC = 0-31. No effect. |
| | | 1 | Corresponding bit in the interrupt pending register (IPR) is cleared (<i>In</i> = 0). |

Figure 4-57. Interrupt Clear Register High (ICRH)

| | | | | | | | | | | | | | | | |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| I63 | I62 | I61 | I60 | I59 | I58 | I57 | I56 | I55 | I54 | I53 | I52 | I51 | I50 | I49 | I48 |
| W-0 |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| I47 | I46 | I45 | I44 | I43 | I42 | I41 | I40 | I39 | I38 | I37 | I36 | I35 | I34 | I33 | I32 |
| W-0 |

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

Table 4-58. Interrupt Clear Register High (ICRH) Field Descriptions

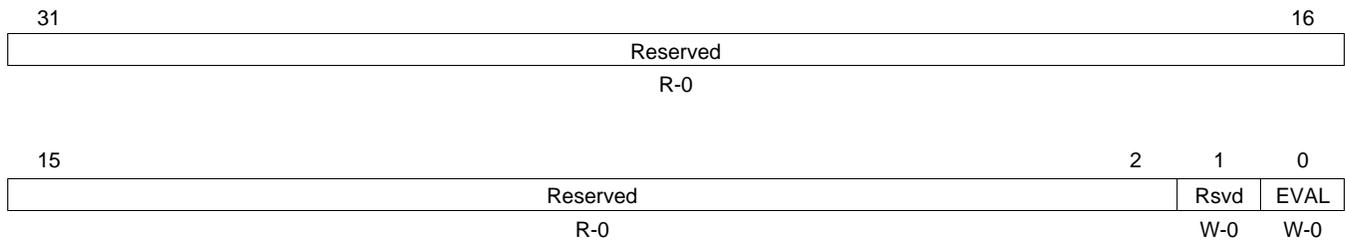
| Bit | Field | Value | Description |
|------|-----------|-------|--|
| 31-0 | <i>In</i> | 0 | Interrupt clear register for TCC = 32-63. No effect. |
| | | 1 | Corresponding bit in the interrupt pending register high (IPRH) is cleared (<i>In</i> = 0). |

4.2.7.6 Interrupt Evaluate Register (IEVAL)

The interrupt evaluate register (IEVAL) is the only register that physically exists in both the global region and the shadow regions. In other words, the read/write accessibility for the shadow region IEVAL is not affected by the DMA/QDMA region access registers (DRAEm/DRAEHm, QRAEn/QRAEHn). IEVAL is needed for robust ISR operations to ensure that interrupts are not missed by the CPU.

The IEVAL is shown in [Figure 4-58](#) and described in [Table 4-59](#).

Figure 4-58. Interrupt Evaluate Register (IEVAL)



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

Table 4-59. Interrupt Evaluate Register (IEVAL) Field Descriptions

| Bit | Field | Value | Description |
|------|----------|--------|--|
| 31-2 | Reserved | 0 | Reserved |
| 1 | Reserved | 0 | Reserved. Always write 0. Writes of 1 to this bit are not supported and attempts to do so may result in undefined behavior. |
| 0 | EVAL | 0 1 | Interrupt evaluate 0 No effect. 1 Causes EDMA3CC completion interrupt to be pulsed, if any enabled (IERn/IERHn = 1) interrupts are still pending (IPRn/IPRHn = 1). The EDMA3CC completion interrupt that is pulsed depends on which IEVAL is being exercised. For example, writing to the EVAL bit in IEVAL pulses the global completion interrupt, but writing to the EVAL bit in IEVAL0 pulses the region 0 completion interrupt. |

4.2.8 QDMA Registers

The following sets of registers control the QDMA channels in the EDMA3CC. The QDMA channels (with the exception of the QDMA queue number register) consist of a set of registers, each of which have a bit location. Each bit position corresponds to a QDMA channel number. The QDMA channel registers are accessible via read/writes to the global address range. They are also accessible via read/writes to the shadow address range. The read/write accessibility in the shadow region address region is controlled by the QDMA region access registers (QRAEn/QRAEHn). [Section 2.7](#) details shadow region/global region usage.

4.2.8.1 QDMA Event Register (QER)

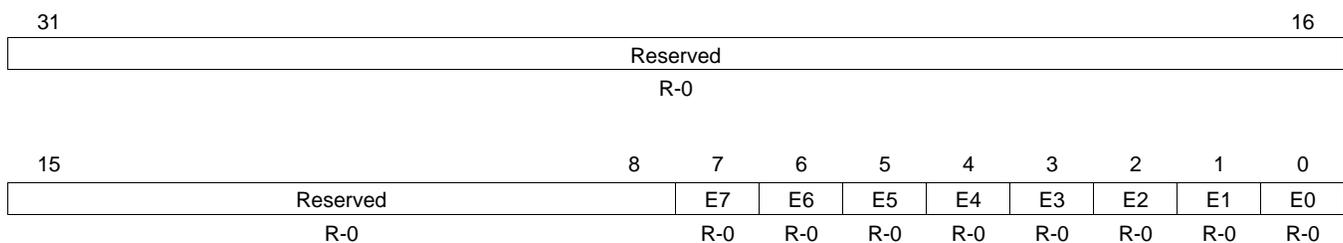
The QDMA event register (QER) channel n bit is set ($En = 1$) when the CPU or any EDMA3 programmer (including EDMA3) performs a write to the trigger word (using the QDMA channel mapping register (QCHMAPn)) in the PaRAM entry associated with QDMA channel n (which is also programmed using QCHMAPn). The En bit is also set when the EDMA3CC performs a link update on a PaRAM address that matches the QCHMAPn settings. The QDMA event is latched only if the QDMA event enable register (QEER) channel n bit is also enabled ($QEER.En = 1$). Once a bit is set in QER, then the corresponding QDMA event (auto-trigger) is evaluated by the EDMA3CC logic for an associated transfer request submission to the transfer controllers. For additional conditions that can lead to the setting of QER bits, see [Section 1.1](#).

The setting of an event is a higher priority relative to clear operations (via hardware). If set and clear conditions occur concurrently, the set condition wins. If the event was previously set, then the QDMA event missed register (QEMR) would be set because an event is lost. If the event was previously clear, then the event remains set and is prioritized for submission to the event queues.

The set bits in QER are only cleared when the transfer request associated with the corresponding channels has been processed by the EDMA3CC and submitted to the transfer controller. If the En bit is already set and a QDMA event for the same QDMA channel occurs prior to the original being cleared, then the second missed event is latched in QEMR ($En = 1$).

The QER is shown in [Figure 4-59](#) and described in [Table 4-60](#).

Figure 4-59. QDMA Event Register (QER)



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

Table 4-60. QDMA Event Register (QER) Field Descriptions

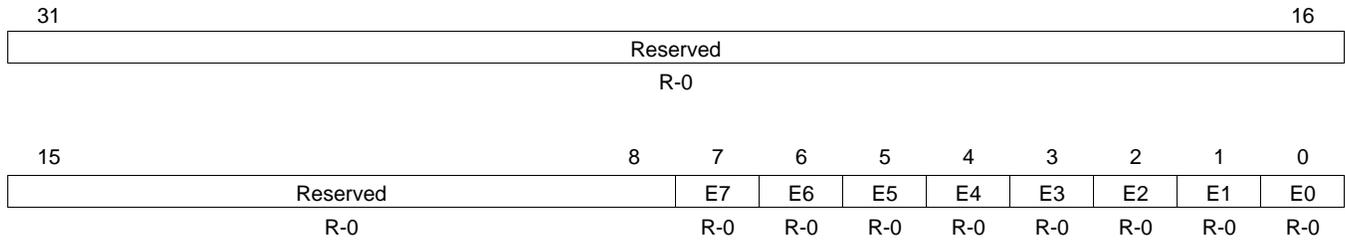
| Bit | Field | Value | Description |
|------|----------|-------|---|
| 31-8 | Reserved | 0 | Reserved |
| 7-0 | En | 0 | No effect. |
| | | 1 | Corresponding QDMA event is prioritized versus other pending DMA/QDMA events for submission to the EDMA3TC. |

4.2.8.2 QDMA Event Enable Register (QEER)

The EDMA3CC provides the option of selectively enabling/disabling each channel in the QDMA event register (QER) by using the QDMA event enable register (QEER). If any of the event bits in QEER is set (using the QDMA event enable set register, QEESR), it enables that corresponding event. Alternatively, if any event bit in QEER is cleared (using the QDMA event enable clear register, QEECR), it disables the corresponding QDMA channel. The QDMA event register does not latch any event for a QDMA channel, if it is not enabled via QEER.

The QEER is shown in [Figure 4-60](#) and described in [Table 4-61](#).

Figure 4-60. QDMA Event Enable Register (QEER)



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

Table 4-61. QDMA Event Enable Register (QEER) Field Descriptions

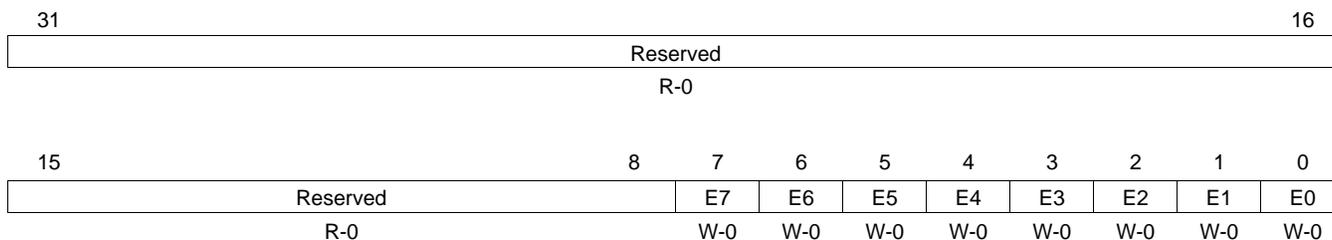
| Bit | Field | Value | Description |
|------|----------|-------|--|
| 31-8 | Reserved | 0 | Reserved |
| 7-0 | E_n | 0 | QDMA event enable for channels 0-7. QDMA channel n is not enabled. QDMA event is not recognized and is not latched in the QDMA event register (QER) |
| | | 1 | QDMA channel n is enabled. QDMA events are recognized and are latched in the QDMA event register (QER) |

4.2.8.3 QDMA Event Enable Clear Register (QEECR)

The QDMA event enable register (QEER) cannot be modified by directly writing to the register, to ease the software burden when multiple tasks are attempting to simultaneously modify these registers. The QDMA event enable clear register (QEECR) is used to disable events. Writes of 1 to the bits in QEECR clear the corresponding QDMA channel bits in QEER; writes of 0 have no effect.

The QEECR is shown in [Figure 4-61](#) and described in [Table 4-62](#).

Figure 4-61. QDMA Event Enable Clear Register (QEECR)



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

Table 4-62. QDMA Event Enable Clear Register (QEECR) Field Descriptions

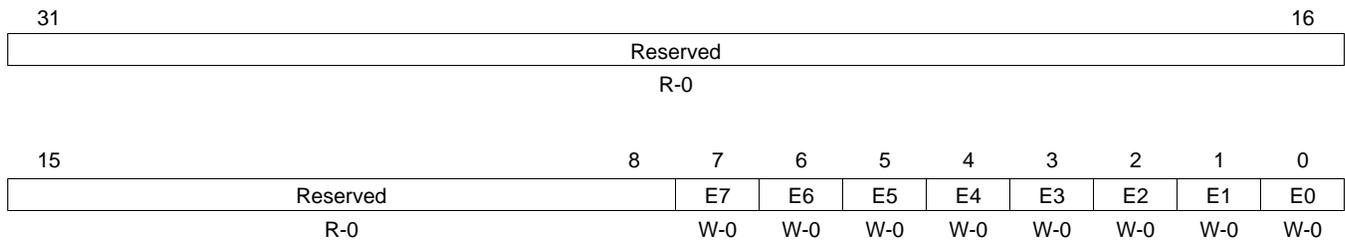
| Bit | Field | Value | Description |
|------|----------|-------|--|
| 31-8 | Reserved | 0 | Reserved |
| 7-0 | E_n | 0 | QDMA event enable clear for channels 0-7. No effect. |
| | | 1 | QDMA event is disabled. Corresponding bit in the QDMA event enable register (QEER) is cleared ($E_n = 0$). |

4.2.8.4 QDMA Event Enable Set Register (QEESR)

The QDMA event enable register (QEER) cannot be modified by directly writing to the register, to ease the software burden when multiple tasks are attempting to simultaneously modify these registers. The QDMA event enable set register (QEESR) is used to enable events. Writes of 1 to the bits in QEESR set the corresponding QDMA channel bits in QEER; writes of 0 have no effect.

The QEESR is shown in [Figure 4-62](#) and described in [Table 4-63](#).

Figure 4-62. QDMA Event Enable Set Register (QEESR)



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

Table 4-63. QDMA Event Enable Set Register (QEESR) Field Descriptions

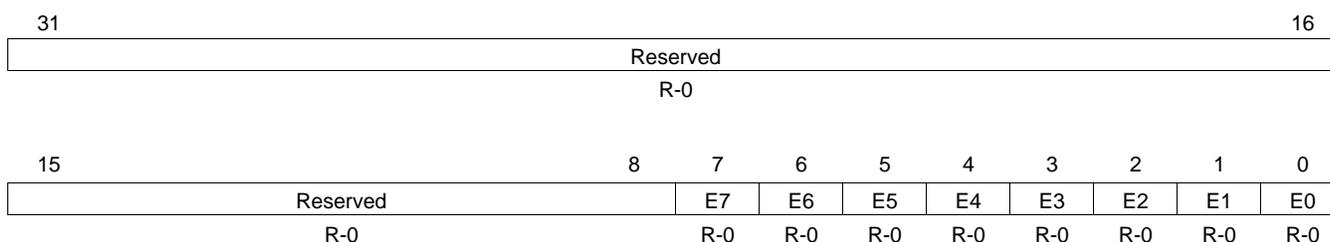
| Bit | Field | Value | Description |
|------|----------|-------|---|
| 31-8 | Reserved | 0 | Reserved |
| 7-0 | E_n | 0 | QDMA event enable set for channels 0-7. No effect. |
| | | 1 | QDMA event is enabled. Corresponding bit in the QDMA event enable register (QEER) is set ($E_n = 1$). |

4.2.8.5 QDMA Secondary Event Register (QSER)

The QDMA secondary event register (QSER) provides information on the state of a QDMA event. If at any time a bit corresponding to a QDMA channel is set in QSER, that implies that the corresponding QDMA event is in the queue. Once a bit corresponding to a QDMA channel is set in QSER, the EDMA3CC does not prioritize additional events on the same QDMA channel. Depending on the condition that lead to the setting of the QSER bits, either the EDMA3CC hardware or the software (using QSECR) needs to clear the QSER bits for the EDMA3CC to evaluate subsequent QDMA events on the channel. Based on whether the associated TR request is valid, or it is a null or dummy TR, the implications on the state of QSER and the required user actions to submit another QDMA transfer might be different. For additional conditions that can cause the secondary event registers (QSER\SER) to be set, see [Section 1.1](#).

The QSER is shown in [Figure 4-63](#) and described in [Table 4-64](#).

Figure 4-63. QDMA Secondary Event Register (QSER)



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

Table 4-64. QDMA Secondary Event Register (QSER) Field Descriptions

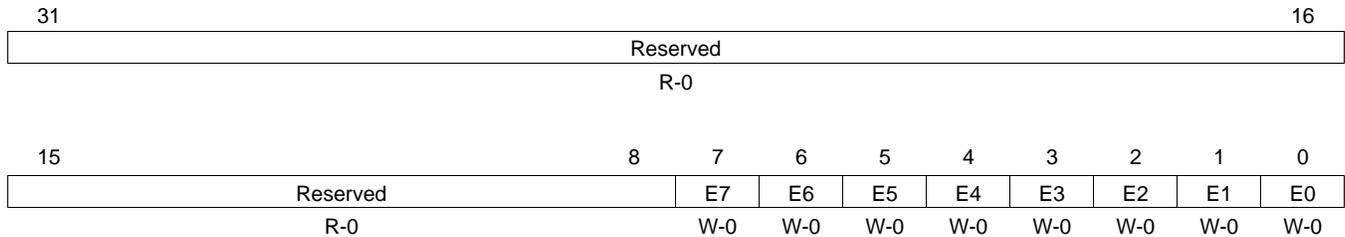
| Bit | Field | Value | Description |
|------|----------|-------|---|
| 31-8 | Reserved | 0 | Reserved |
| 7-0 | E_n | 0 | QDMA secondary event register for channels 0-7. QDMA event is not currently stored in the event queue. |
| | | 1 | QDMA event is currently stored in the event queue. EDMA3CC does not prioritize additional events. |

4.2.8.6 QDMA Secondary Event Clear Register (QSECR)

The QDMA secondary event clear register (QSECR) clears the status of the QDMA secondary event register (QSER) and the QDMA event register (QER). CPU writes of 1 clear the corresponding set bits in QSER and QER. Writes of 0 have no effect. Note that this differs from the secondary event clear register (SECR) operation, which only clears the secondary event register (SER) bits and does not affect the event registers.

The QSECR is shown in [Figure 4-64](#) and described in [Table 4-65](#).

Figure 4-64. QDMA Secondary Event Clear Register (QSECR)



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

Table 4-65. QDMA Secondary Event Clear Register (QSECR) Field Descriptions

| Bit | Field | Value | Description |
|------|----------|-------|--|
| 31-8 | Reserved | 0 | Reserved |
| 7-0 | En | 0 | No effect. |
| | | 1 | Corresponding bit in the QDMA secondary event register (QSER) and the QDMA event register (QER) is cleared (En = 0). |

4.3 EDMA3 Transfer Controller Control Registers

Table 4-66 lists the memory-mapped registers for the EDMA3 transfer controller (EDMA3TC). For the memory addresses of these registers, see the device-specific data manual.

Table 4-66. EDMA3 Transfer Controller Registers

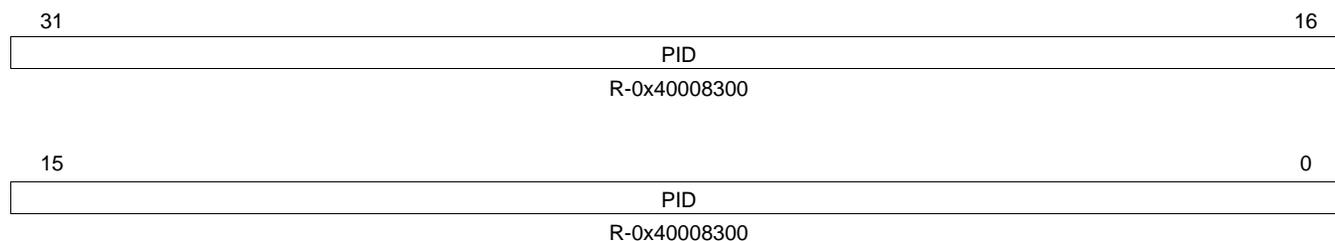
| Offset | Acronym | Register Description | See |
|--------|-----------|---|----------------------------------|
| 00h | PID | Peripheral Identification Register | Section 4.3.1 |
| 04h | TCCFG | EDMA3TC Configuration Register | Section 4.3.2 |
| 0100h | TCSTAT | EDMA3TC Channel Status Register | Section 4.3.3 |
| 0120h | ERRSTAT | Error Register | Section 4.3.4.1 |
| 0124h | ERREN | Error Enable Register | Section 4.3.4.2 |
| 0128h | ERRCLR | Error Clear Register | Section 4.3.4.3 |
| 012Ch | ERRDET | Error Details Register | Section 4.3.4.4 |
| 0130h | ERRCMD | Error Interrupt Command Register | Section 4.3.4.5 |
| 0140h | RDRATE | Read Rate Register | Section 4.3.5 |
| 0240h | SAOPT | Source Active Options Register | Section 4.3.6.1 |
| 0244h | SASRC | Source Active Source Address Register | Section 4.3.6.2 |
| 0248h | SACNT | Source Active Count Register | Section 4.3.6.3 |
| 024Ch | SADST | Source Active Destination Address Register | Section 4.3.6.4 |
| 0250h | SABIDX | Source Active Source B-Index Register | Section 4.3.6.5 |
| 0254h | SAMPPRXY | Source Active Memory Protection Proxy Register | Section 4.3.6.6 |
| 0258h | SACNTRL | Source Active Count Reload Register | Section 4.3.6.7 |
| 025Ch | SASRCBREF | Source Active Source Address B-Reference Register | Section 4.3.6.8 |
| 0260h | SADSTBREF | Source Active Destination Address B-Reference Register | Section 4.3.6.9 |
| 0280h | DFCNTRL | Destination FIFO Set Count Reload | Section 4.3.6.16 |
| 0284h | DFSRCBREF | Destination FIFO Set Destination Address B Reference Register | Section 4.3.6.17 |
| 0288h | DFDSTBREF | Destination FIFO Set Destination Address B Reference Register | Section 4.3.6.18 |
| 0300h | DFOPT0 | Destination FIFO Options Register 0 | Section 4.3.6.10 |
| 0304h | DFSRC0 | Destination FIFO Source Address Register 0 | Section 4.3.6.11 |
| 0308h | DFCNT0 | Destination FIFO Count Register 0 | Section 4.3.6.12 |
| 030Ch | DFDST0 | Destination FIFO Destination Address Register 0 | Section 4.3.6.13 |
| 0310h | DFBIDX0 | Destination FIFO BIDX Register 0 | Section 4.3.6.14 |
| 0314h | DFMPPRXY0 | Destination FIFO Memory Protection Proxy Register 0 | Section 4.3.6.15 |
| 0340h | DFOPT1 | Destination FIFO Options Register 1 | Section 4.3.6.10 |
| 0344h | DFSRC1 | Destination FIFO Source Address Register 1 | Section 4.3.6.11 |
| 0348h | DFCNT1 | Destination FIFO Count Register 1 | Section 4.3.6.12 |
| 034Ch | DFDST1 | Destination FIFO Destination Address Register 1 | Section 4.3.6.13 |
| 0350h | DFBIDX1 | Destination FIFO BIDX Register 1 | Section 4.3.6.14 |
| 0354h | DFMPPRXY1 | Destination FIFO Memory Protection Proxy Register 1 | Section 4.3.6.15 |
| 0380h | DFOPT2 | Destination FIFO Options Register 2 | Section 4.3.6.10 |
| 0384h | DFSRC2 | Destination FIFO Source Address Register 2 | Section 4.3.6.11 |
| 0388h | DFCNT2 | Destination FIFO Count Register 2 | Section 4.3.6.12 |
| 038Ch | DFDST2 | Destination FIFO Destination Address Register 2 | Section 4.3.6.13 |
| 0390h | DFBIDX2 | Destination FIFO BIDX Register 2 | Section 4.3.6.14 |
| 0394h | DFMPPRXY2 | Destination FIFO Memory Protection Proxy Register 2 | Section 4.3.6.15 |
| 03C0h | DFOPT3 | Destination FIFO Options Register 3 | Section 4.3.6.10 |
| 03C4h | DFSRC3 | Destination FIFO Source Address Register 3 | Section 4.3.6.11 |
| 03C8h | DFCNT3 | Destination FIFO Count Register 3 | Section 4.3.6.12 |

Table 4-66. EDMA3 Transfer Controller Registers (continued)

| Offset | Acronym | Register Description | See |
|--------|-----------|---|----------------------------------|
| 03CCh | DFDST3 | Destination FIFO Destination Address Register 3 | Section 4.3.6.13 |
| 03D0h | DFBIDX3 | Destination FIFO BIDX Register 3 | Section 4.3.6.14 |
| 03D4h | DFMPPRXY3 | Destination FIFO Memory Protection Proxy Register 3 | Section 4.3.6.15 |

4.3.1 Peripheral Identification Register (PID)

The peripheral identification register (PID) is a constant register that uniquely identifies the EDMA3TC and specific revision of the EDMA3TC. The PID is shown in [Figure 4-65](#) and described in [Table 4-67](#).

Figure 4-65. Peripheral ID Register (PID)


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

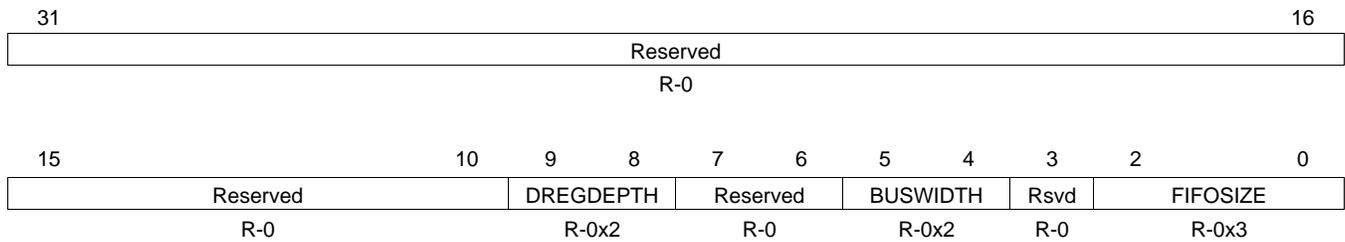
Table 4-67. Peripheral ID Register (PID) Field Descriptions

| Bit | Field | Value | Description |
|------|-------|--------------|------------------------|
| 31-0 | PID | 0-FFFF FFFFh | Peripheral identifier. |

4.3.2 EDMA3TC Configuration Register (TCCFG)

The EDMA3TC configuration register (TCCFG) is shown in [Figure 4-66](#) and described in [Table 4-68](#).

Figure 4-66. EDMA3TC Configuration Register (TCCFG)



LEGEND: R/W = Read/Write; R = Read only; -n = value after reset; -x = value is indeterminate after reset

Table 4-68. EDMA3TC Configuration Register (TCCFG) Field Descriptions

| Bit | Field | Value | Description |
|-------|-----------|-------|---|
| 31-10 | Reserved | 0 | Reserved |
| 9-8 | DREGDEPTH | 0-3h | Destination register FIFO depth parameterization. |
| | | 0-1h | Reserved |
| | | 2h | 4 entry (for TC0, TC1, TC2, TC3, TC4, and TC5) |
| | | 3h | Reserved |
| 7-6 | Reserved | 0 | Reserved |
| 5-4 | BUSWIDTH | 0-3h | Bus width parameterization. |
| | | 0-1h | Reserved |
| | | 2h | 128-bit (for TC0, TC1, TC2, TC3, TC4, and TC5) |
| | | 3h | Reserved |
| 3 | Reserved | 0 | Reserved |
| 2-0 | FIFOSIZE | 0-2h | Reserved |
| | | 3h | 256 byte FIFO (for TC0, TC1, TC2, TC3, TC4 and TC5) |
| | | 4h-7h | Reserved |

4.3.3 EDMA3TC Channel Status Register (TCSTAT)

The EDMA3TC channel status register (TCSTAT) is shown in [Figure 4-67](#) and described in [Table 4-69](#).

Figure 4-67. EDMA3TC Channel Status Register (TCSTAT)

| | | | | | | |
|----------|----------|-----------|----------|----------|----------|----------|
| | Reserved | | | | | |
| | R-0 | | | | | |
| 15 | 14 | 13 | 12 | 11 | 9 | 8 |
| Reserved | Reserved | DFSTRTPTR | Reserved | Reserved | Reserved | Reserved |
| R-0 | R-0 | R-0 | R-0 | R-0 | R-0 | R-1 |
| 7 | 6 | 4 | 3 | 2 | 1 | 0 |
| Reserved | DSTACTV | Reserved | WSACTV | SRCACTV | PROGBUSY | Reserved |
| R-0 | R-0 | R-0 | R-0 | R-0 | R-0 | R-0 |

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

Table 4-69. EDMA3TC Channel Status Register (TCSTAT) Field Descriptions

| Bit | Field | Value | Description |
|-------|-----------|----------|---|
| 31-14 | Reserved | 0 | Reserved |
| 13-12 | DFSTRTPTR | 0-3h | Destination FIFO start pointer. Represents the offset to the head entry of the destination register FIFO, in units of entries. 0 For TC0, TC1,TC2 ,TC3, TC4 and TC5: if the destination register FIFO depth parameterization is 4. 1h For TC0, TC1,TC2 ,TC3, TC4 and TC5: if the destination register FIFO depth parameterization is 4. 2h For TC0, TC1,TC2 ,TC3, TC4 and TC5: if the destination register FIFO depth parameterization is 4. 3h For TC0, TC1,TC2 ,TC3, TC4 and TC5: if the destination register FIFO depth parameterization is 4. |
| 11-9 | Reserved | 0 | Reserved |
| 8 | Reserved | 1 | Reserved. Always read as 1. |
| 7 | Reserved | 0 | Reserved |
| 6-4 | DSTACTV | 0-7h | Destination active state. Specifies the number of transfer requests (TRs) that are resident in the destination register FIFO at a given instant. This bit field can be primarily used for advanced debugging. Legal values are constrained by the destination register FIFO depth parameterization (DSTREGDEPTH) parameter. 0 Destination FIFO is empty. 1h Destination FIFO contains 1 TR. 2h Destination FIFO contains 2 TRs. 3h Destination FIFO contains 3 TRs. 4h Destination FIFO contains 4 TRs. (Full if DSTREGDEPTH==4) If the destination register FIFO is empty, then any TR written to Prog Set immediately transitions to the destination register FIFO. If the destination register FIFO is not empty and not full, then any TR written to Prog Set immediately transitions to the destination register FIFO set if the source active state (SRCACTV) bit is set to idle. If the destination register FIFO is full, then TRs cannot transition to the destination register FIFO. The destination register FIFO becomes not full when the TR at the head of the destination register FIFO is completed. |
| 5h-7h | | Reserved | Reserved |
| 3 | Reserved | 0 | Reserved |
| 2 | WSACTV | 0 1 | Write status active. 0 Write status is not pending. Write status has been received for all previously issued write commands. 1 Write status is pending. Write status has not been received for all previously issued write commands. |

Table 4-69. EDMA3TC Channel Status Register (TCSTAT) Field Descriptions (continued)

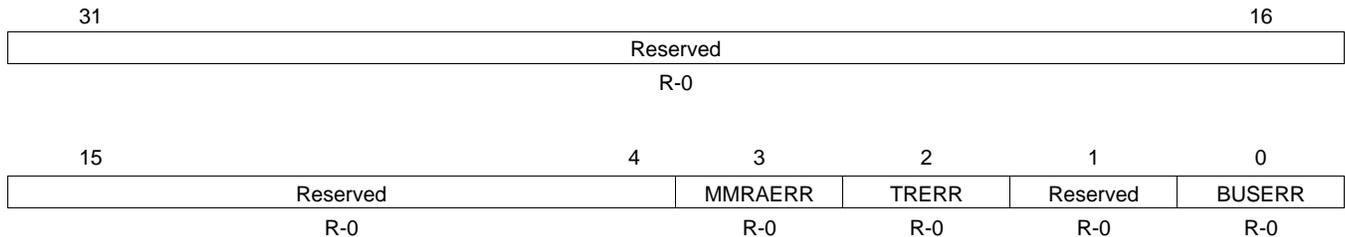
| Bit | Field | Value | Description |
|-----|----------|-------|---|
| 1 | SRCACTV | 0 | Source controller is idle. Source active register set contains a previously processed transfer request. |
| | | 1 | Source controller is busy servicing a transfer request. |
| 0 | PROGBUSY | 0 | Program set idle and is available for programming by the EDMA3CC. |
| | | 1 | Program set busy. |

4.3.4 Error Registers

4.3.4.1 Error Register (ERRSTAT)

The error status register (ERRSTAT) is shown in [Figure 4-68](#) and described in [Table 4-70](#).

Figure 4-68. Error Register (ERRSTAT)



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

Table 4-70. Error Register (ERRSTAT) Field Descriptions

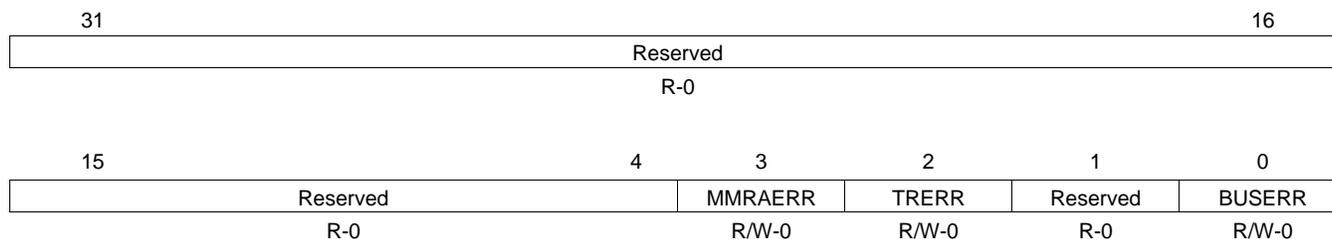
| Bit | Field | Value | Description |
|------|----------|-------|---|
| 31-4 | Reserved | 0 | Reserved |
| 3 | MMRAERR | 0 | MMR address error. Condition is not detected |
| | | 1 | User attempted to read or write to an invalid address in configuration memory map. |
| 2 | TRERR | 0 | Transfer request (TR) error event. Condition is not detected. |
| | | 1 | TR detected that violates FIFO mode transfer (SAM or DAM is set) alignment rules or has ACNT or BCNT == 0. |
| 1 | Reserved | 0 | Reserved |
| 0 | BUSERR | 0 | Bus error event. Condition is not detected. |
| | | 1 | EDMA3TC has detected an error at source or destination address. Error information can be read from the error details register (ERRDET). |

4.3.4.2 Error Enable Register (ERREN)

The error enable register (ERREN) is shown in [Figure 4-69](#) and described in [Table 4-71](#).

When any of the enable bits are set, a bit set in the corresponding ERRSTAT causes an assertion of the EDMA3TC interrupt.

Figure 4-69. Error Enable Register (ERREN)



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

Table 4-71. Error Enable Register (ERREN) Field Descriptions

| Bit | Field | Value | Description |
|------|----------|-------|---|
| 31-4 | Reserved | 0 | Reserved |
| 3 | MMRAERR | 0 | Interrupt enable for MMR address error (MMRAERR). MMRAERR is disabled. |
| | | 1 | MMRAERR is enabled and contributes to the state of EDMA3TC error interrupt generation |
| 2 | TRERR | 0 | Interrupt enable for transfer request error (TRERR). TRERR is disabled. |
| | | 1 | TRERR is enabled and contributes to the state of EDMA3TC error interrupt generation. |
| 1 | Reserved | | Reserved |
| 0 | BUSERR | 0 | Interrupt enable for bus error (BUSERR). BUSERR is disabled. |
| | | 1 | BUSERR is enabled and contributes to the state of EDMA3TC error interrupt generation. |

4.3.4.3 Error Clear Register (ERRCLR)

The error clear register (ERRCLR) is shown in [Figure 4-70](#) and described in [Table 4-72](#).

Figure 4-70. Error Clear Register (ERRCLR)



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

Table 4-72. Error Clear Register (ERRCLR) Field Descriptions

| Bit | Field | Value | Description |
|------|----------|-------|--|
| 31-4 | Reserved | 0 | Reserved |
| 3 | MMRAERR | 0 | Interrupt enable clear for the MMRAERR bit in the error status register (ERRSTAT). No effect. |
| | | 1 | Clears the MMRAERR bit in ERRSTAT but does not clear the error details register (ERRDET). |
| 2 | TRERR | 0 | Interrupt enable clear for the TRERR bit in the error status register (ERRSTAT). No effect. |
| | | 1 | Clears the TRERR bit in ERRSTAT but does not clear the error details register (ERRDET). |
| 1 | Reserved | 0 | Reserved |
| 0 | BUSERR | 0 | Interrupt clear for the BUSERR bit in the error status register (ERRSTAT). No effect. |
| | | 1 | Clears the BUSERR bit in ERRSTAT and clears the error details register (ERRDET). |

4.3.4.4 Error Details Register (ERRDET)

The error details register (ERRDET) is shown in [Figure 4-71](#) and described in [Table 4-73](#).

Figure 4-71. Error Details Register (ERRDET)

| | | | | | | | | | | | | | | |
|----------|--|--|-----|----|----|--|--|----------|---|--|--------|---------|----|---|
| 31 | | | | | | | | | | | 18 | 17 | 16 | |
| Reserved | | | | | | | | | | | TCCHEN | TCINTEN | | |
| R-0 | | | | | | | | | | | R-0 | R-0 | | |
| | | | 15 | 14 | 13 | | | 8 | 7 | | | 4 | 3 | 0 |
| Reserved | | | TCC | | | | | Reserved | | | STAT | | | |
| R-0 | | | R-0 | | | | | R-0 | | | R-0 | | | |

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

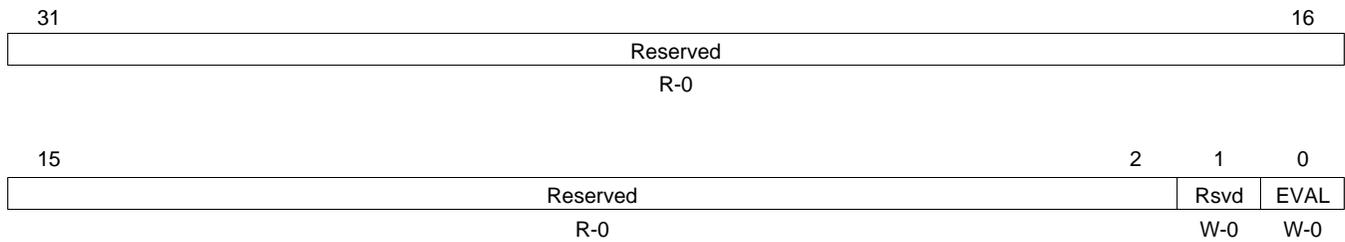
Table 4-73. Error Details Register (ERRDET) Field Descriptions

| Bit | Field | Value | Description |
|--------|----------|-------|--|
| 31-8 | Reserved | 0 | Reserved |
| 17 | TCCHEN | 0-1 | Transfer completion chaining enable. Contains the TCCHEN value in the channel options parameter (OPT) programmed by the channel controller for the read or write transaction that resulted in an error. |
| 16 | TCINTEN | 0-1 | Transfer completion interrupt enable. Contains the TCINTEN value in the channel options parameter (OPT) programmed by the channel controller for the read or write transaction that resulted in an error. |
| 15-14 | Reserved | 0 | Reserved |
| 13 - 8 | TCC | 0-3Fh | Transfer complete code. Contains the TCC value in the channel options parameter (OPT) programmed by the channel controller for the read or write transaction that resulted in an error. |
| 7-4 | Reserved | 0 | Reserved |
| 3-0 | STAT | 0-Fh | Transaction status. Stores the nonzero status/error code that was detected on the read status or write status bus. If read status and write status are returned on the same cycle, then the EDMA3TC chooses nonzero version. If both are nonzero, then the write status is treated as higher priority. |
| | | 0 | No error |
| | | 1h-7h | Read error |
| | | 8h-Fh | Write error |

4.3.4.5 Error Interrupt Command Register (ERRCMD)

The error command register (ERRCMD) is shown in [Figure 4-72](#) and described in [Table 4-74](#).

Figure 4-72. Error Interrupt Command Register (ERRCMD)



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

Table 4-74. Error Interrupt Command Register (ERRCMD) Field Descriptions

| Bit | Field | Value | Description |
|------|----------|-------|--|
| 31-2 | Reserved | 0 | Reserved |
| 1 | Reserved | 0 | Reserved. Writes of 1 to this bit are not supported. Attempts to do so may result in undefined behavior. |
| 0 | EVAL | 0 | Error evaluate. No effect. |
| | | 1 | EDMA3TC error line is pulsed if any of the error status register (ERRSTAT) bits are set. |

4.3.5 Read Rate Register (RDRATE)

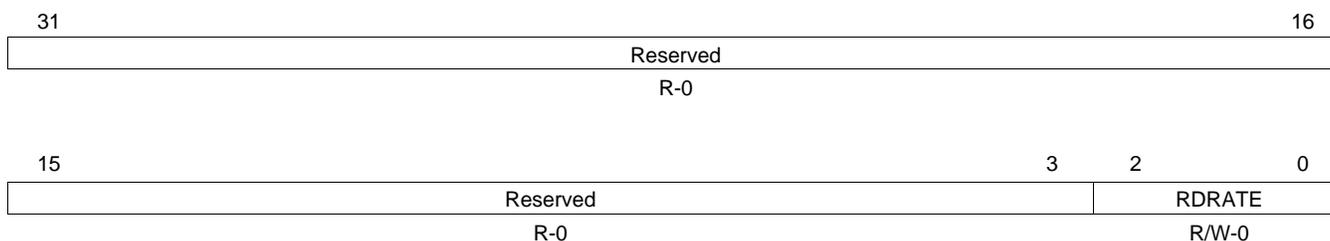
The EDMA3 transfer controller issues read commands at a rate controlled by the read rate register (RDRATE). The RDRATE defines the number of idle cycles that the read controller must wait before issuing subsequent commands. This applies both to commands within a transfer request packet (TRP) and for commands that are issued for different transfer requests (TRs). For instance, if RDRATE is set to 4 cycles between reads, there are 3 inactive cycles between reads.

RDRATE allows flexibility in transfer controller access requests to an endpoint. For an application, RDRATE can be manipulated to slow down the access rate, so that the endpoint may service requests from other masters during the inactive EDMA3TC cycles.

The RDRATE is shown in [Figure 4-73](#) and described in [Table 4-75](#).

Note: It is expected that the RDRATE value for a transfer controller is static, as it is decided based on the application requirement. It is not recommended to change this setting on the fly.

Figure 4-73. Read Rate Register (RDRATE)



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

Table 4-75. Read Rate Register (RDRATE) Field Descriptions

| Bit | Field | Value | Description |
|------|----------|-------|--|
| 31-3 | Reserved | 0 | Reserved |
| 2-0 | RDRATE | 0-7h | Read rate. Controls the number of cycles between read commands. This is a global setting that applies to all TRs for this EDMA3TC. |
| | | 0 | Reads issued as fast as possible. |
| | | 1h | 4 cycles between reads. |
| | | 2h | 8 cycles between reads. |
| | | 3h | 16 cycles between reads. |
| | | 4h | 32 cycles between reads. |
| | | 5h-7h | Reserved |

4.3.6 EDMA3TC Channel Registers

The EDMA3TC channel registers are split into three parts: the programming registers, the source active registers, and the destination FIFO register. This section describes the registers and their functions. The program register set is programmed by the channel controller, and is for internal use. The other two sets are read-only and provided to facilitate advanced debug capabilities. The number of destination FIFO register sets depends on the destination FIFO depth.

TC0, TC1, TC2, TC3, TC4 and TC5 all have a destination FIFO depth of 4, so there are four sets of destination FIFO registers for each of these transfer controllers.

4.3.6.1 Source Active Options Register (SAOPT)

The source active options register (SAOPT) is shown in [Figure 4-74](#) and described in [Table 4-76](#).

Figure 4-74. Source Active Options Register (SAOPT)

| | | | | | | | | | | | | | | |
|-------|----------|--|------|-------|---|------|--------|-------|---------|----------|-----|-------|-------|----|
| 31 | Reserved | | | | | | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | | | | | | | TCCHEN | Rsvd | TCINTEN | Reserved | | TCC | | |
| | | | | | | | R-0 | R/W-0 | R-0 | R/W-0 | R-0 | | R/W-0 | |
| 15 | 12 | | 11 | 10 | 8 | 7 | 6 | 4 | | 3 | 2 | 1 | 0 | |
| TCC | | | Rsvd | FWID | | Rsvd | PRI | | | Reserved | | DAM | SAM | |
| R/W-0 | | | R-0 | R/W-0 | | R-0 | R/W-0 | | | R-0 | | R/W-0 | R/W-0 | |

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

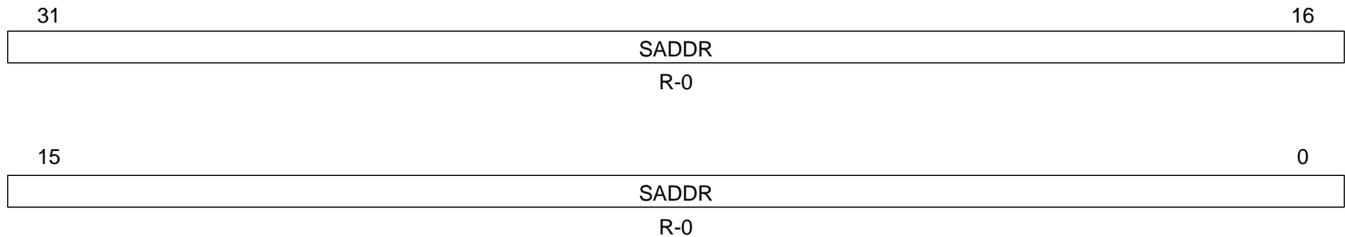
Table 4-76. Source Active Options Register (SAOPT) Field Descriptions

| Bit | Field | Value | Description |
|-------|----------|--|---|
| 31-23 | Reserved | 0 | Reserved |
| 22 | TCCHEN | 0 1 | Transfer complete chaining enable. 0 Transfer complete chaining is disabled. 1 Transfer complete chaining is enabled. |
| 21 | Reserved | 0 | Reserved |
| 20 | TCINTEN | 0 1 | Transfer complete interrupt enable. 0 Transfer complete interrupt is disabled. 1 Transfer complete interrupt is enabled. |
| 19-18 | Reserved | 0 | Reserved |
| 17-12 | TCC | 0-3Fh | Transfer complete code. This 6-bit code is used to set the relevant bit in CER or IPR of the EDMA3PCC module. |
| 11 | Reserved | 0 | Reserved |
| 10-8 | FWID | 0-7h 0 1h 2h 3h 4h 5h 6h-7h | FIFO width. Applies if either SAM or DAM is set to FIFO mode. 0 FIFO width is 8-bit. 1h FIFO width is 16-bit. 2h FIFO width is 32-bit. 3h FIFO width is 64-bit. 4h FIFO width is 128-bit. 5h FIFO width is 256-bit. 6h-7h Reserved |
| 7 | Reserved | 0 | Reserved |
| 6-4 | PRI | 0-7h 0 1h-6h 7h | Transfer priority. Reflects the values programmed in the QUEPRI register in the EDMACC. 0 Priority 0 - Highest priority 1h-6h Priority 1 to priority 6 7h Priority 7 - Lowest priority |
| 3-2 | Reserved | 0 | Reserved |
| 1 | DAM | 0 1 | Destination address mode within an array. 0 INCR: Destination addressing within an array increments. 1 FIFO: Destination addressing within an array wraps around upon reaching FIFO width. |
| 0 | SAM | 0 1 | Source address mode within an array. 0 INCR: Source addressing within an array increments. 1 FIFO: Source addressing within an array wraps around upon reaching FIFO width. |

4.3.6.2 Source Active Source Address Register (SASRC)

The source active source address register (SASRC) is shown in [Figure 4-75](#) and described in [Table 4-77](#).

Figure 4-75. Source Active Source Address Register (SASRC)



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

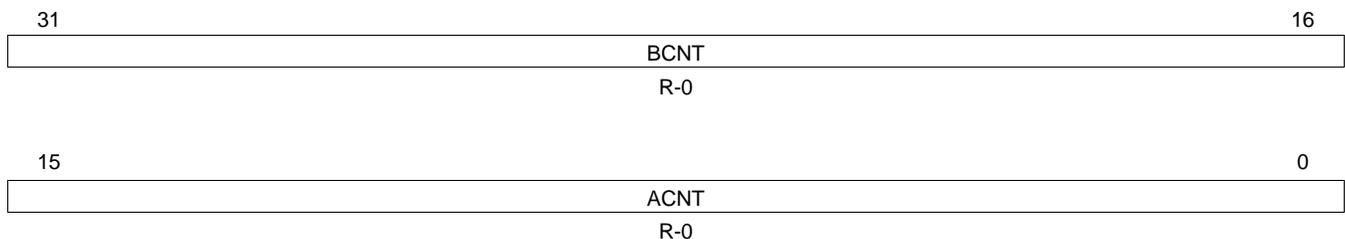
Table 4-77. Source Active Source Address Register (SASRC) Field Descriptions

| Bit | Field | Value | Description |
|------|-------|--------------|---|
| 31-0 | SADDR | 0-FFFF FFFFh | Source address for program register set. EDMA3TC updates value according to source addressing mode (SAM bit in the source active options register, SAOPT) . |

4.3.6.3 Source Active Count Register (SACNT)

The source active count register (SACNT) is shown in [Figure 4-76](#) and described in [Table 4-78](#).

Figure 4-76. Source Active Count Register (SACNT)



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

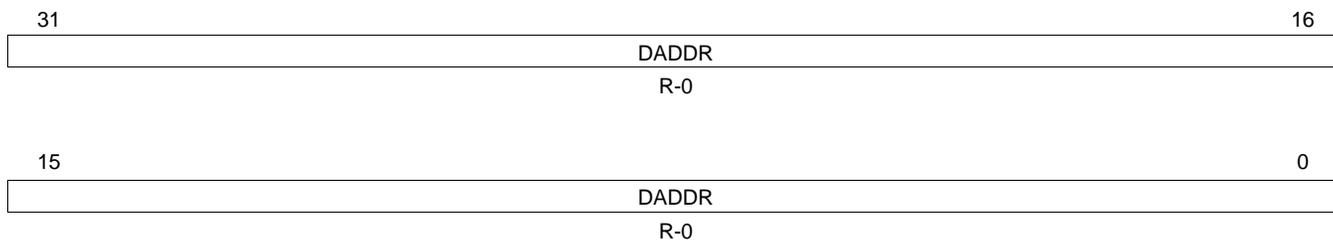
Table 4-78. Source Active Count Register (SACNT) Field Descriptions

| Bit | Field | Value | Description |
|-------|-------|---------|--|
| 31-16 | BCNT | 0-FFFFh | B dimension count. Number of arrays to be transferred, where each array is ACNT in length. It is decremented after each read command appropriately. Represents the amount of data remaining to be read. It should be 0 when transfer request (TR) is complete. |
| 15-0 | ACNT | 0-FFFFh | A dimension count. Number of bytes to be transferred in first dimension. It is decremented after each read command appropriately. Represents the amount of data remaining to be read. It should be 0 when transfer request (TR) is complete. |

4.3.6.4 Source Active Destination Address Register (SADST)

The source active destination address register (SADST) is shown in [Figure 4-77](#) and described in [Table 4-79](#).

Figure 4-77. Source Active Destination Address Register (SADST)



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

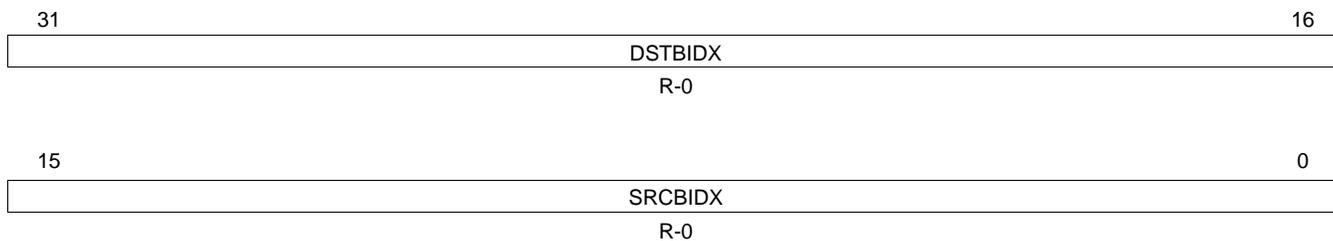
Table 4-79. Source Active Destination Address Register (SADST) Field Descriptions

| Bit | Field | Value | Description |
|------|-------|-------|-------------------|
| 31-0 | DADDR | 0 | Always reads as 0 |

4.3.6.5 Source Active Source B-Dimension Index Register (SABIDX)

The source active set B-dimension index register (SABIDX) is shown in [Figure 4-78](#) and described in [Table 4-80](#).

Figure 4-78. Source Active Source B-Dimension Index Register (SABIDX)



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

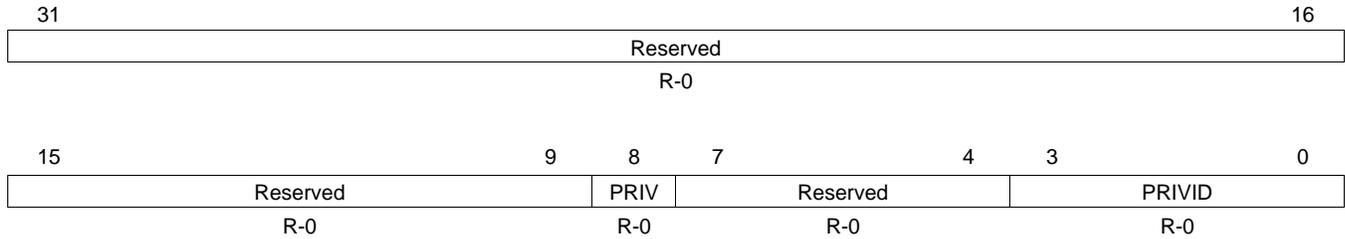
Table 4-80. Source Active Source B-Dimension Index Register (SABIDX) Field Descriptions

| Bit | Field | Value | Description |
|-------|---------|---------|--|
| 31-16 | DSTBIDX | 0 | B-Index offset between destination arrays. Represents the offset in bytes between the starting address of each destination. Always reads as 0. |
| 15-0 | SRCBIDX | 0-FFFFh | B-Index offset between source arrays. Represents the offset in bytes between the starting address of each source array. |

4.3.6.6 Source Active Memory Protection Proxy Register (SAMPPrXY)

The source active memory protection proxy register (SAMPPrXY) is shown in [Figure 4-79](#) and described in [Table 4-81](#).

Figure 4-79. Source Active Memory Protection Proxy Register (SAMPPrXY)



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

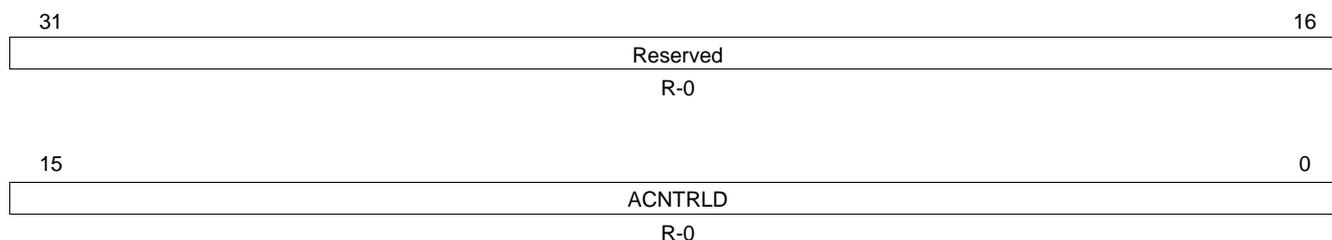
Table 4-81. Source Active Memory Protection Proxy Register (SAMPPrXY) Field Descriptions

| Bit | Field | Value | Description |
|------|----------|--------|---|
| 31-9 | Reserved | 0 | Reserved |
| 8 | PRIV | 0 1 | Privilege level. The privilege level used by the host to set up the parameter entry in the channel controller. This field is set up when the associated TR is submitted to the EDMA3TC. The privilege ID is used while issuing read and write command to the target endpoints so that the target endpoints can perform memory protection checks based on the PRIV of the host that set up the DMA transaction. 0 User-level privilege 1 Supervisor-level privilege |
| 7-4 | Reserved | 0 | Reserved |
| 3-0 | PRIVID | 0-Fh | Privilege ID. This contains the privilege ID of the host that set up the parameter entry in the channel controller. This field is set up when the associated TR is submitted to the EDMA3TC. This PRIVID value is used while issuing read and write commands to the target endpoints so that the target endpoints can perform memory protection checks based on the PRIVID of the host that set up the DMA transaction. |

4.3.6.7 Source Active Count Reload Register (SACNTRLD)

The source active count reload register (SACNTRLD) is shown in [Figure 4-80](#) and described in [Table 4-82](#).

Figure 4-80. Source Active Count Reload Register (SACNTRLD)



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

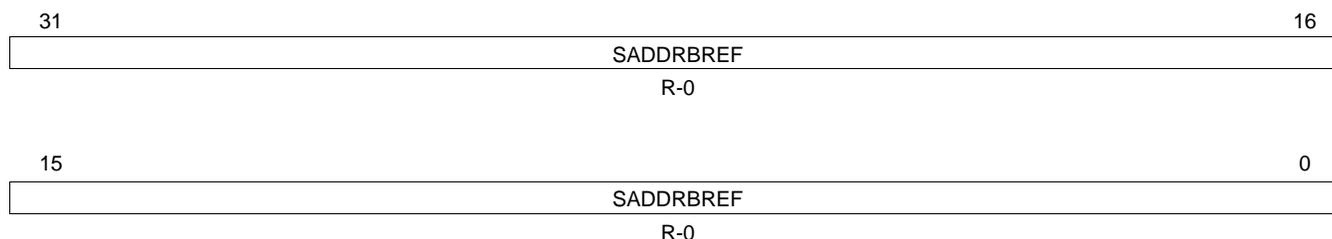
Table 4-82. Source Active Count Reload Register (SACNTRLD) Field Descriptions

| Bit | Field | Value | Description |
|-------|----------|---------|---|
| 31-16 | Reserved | 0 | Reserved |
| 15-0 | ACNTRLD | 0-FFFFh | A-count reload value. Represents the originally programmed value of ACNT. The reload value is used to reinitialize ACNT after each array is serviced. |

4.3.6.8 Source Active Source Address B-Reference Register (SASRCBREF)

The source active source address B-reference register (SASRCBREF) is shown in [Figure 4-81](#) and described in [Table 4-83](#).

Figure 4-81. Source Active Source Address B-Reference Register (SASRCBREF)



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

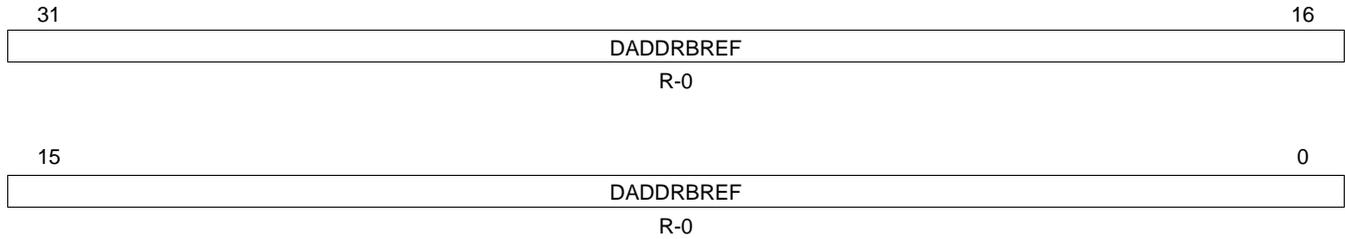
Table 4-83. Source Active Source Address B-Reference Register (SASRCBREF) Field Descriptions

| Bit | Field | Value | Description |
|------|-----------|--------------|---|
| 31-0 | SADDRBREF | 0-FFFF FFFFh | Source address B-reference. Represents the starting address for the array currently being read. |

4.3.6.9 Source Active Destination Address B-Reference Register (SADSTBREF)

The source active destination address B-reference register (SADSTBREF) is shown in [Figure 4-82](#) and described in [Table 4-84](#).

Figure 4-82. Source Active Destination Address B-Reference Register (SADSTBREF)



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

Table 4-84. Source Active Destination Address B-Reference Register (SADSTBREF) Field Descriptions

| Bit | Field | Value | Description |
|------|-----------|-------|-------------------|
| 31-0 | DADDRBREF | 0 | Always reads as 0 |

4.3.6.10 Destination FIFO Options Register (DFOPT_n)

The destination FIFO options register (DFOPT_n) is shown in [Figure 4-83](#) and described in [Table 4-85](#).

Note: The value for *n* varies from 0 to DSTREGDEPTH for the given EDMA3TC.

Figure 4-83. Destination FIFO Options Register (DFOPT_n)

| | | | | | | | | | | | | | | | | |
|----------|--|--|-------|----|------|--------|------|---------|----------|----|-------|----------|----|-------|-------|---|
| 31 | | | | | | | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | | |
| Reserved | | | | | | TCCHEN | Rsvd | TCINTEN | Reserved | | TCC | | | | | |
| R-0 | | | | | | R/W-0 | R-0 | R/W-0 | R-0 | | R/W-0 | | | | | |
| | | | 15 | 12 | 11 | 10 | 8 | 7 | 6 | | | 4 | 3 | 2 | 1 | 0 |
| | | | TCC | | Rsvd | FWID | | Rsvd | PRI | | | Reserved | | DAM | SAM | |
| | | | R/W-0 | | R-0 | R/W-0 | | R-0 | R/W-0 | | | R-0 | | R/W-0 | R/W-0 | |

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

Table 4-85. Destination FIFO Options Register (DFOPT_n) Field Descriptions

| Bit | Field | Value | Description |
|-------|----------|--|--|
| 31-23 | Reserved | 0 | Reserved |
| 22 | TCCHEN | 0 1 | Transfer complete chaining enable. Transfer complete chaining is disabled. Transfer complete chaining is enabled. |
| 21 | Reserved | 0 | Reserved |
| 20 | TCINTEN | 0 1 | Transfer complete interrupt enable. Transfer complete interrupt is disabled. Transfer complete interrupt is enabled. |
| 19-18 | Reserved | 0 | Reserved |
| 17-12 | TCC | 0-3Fh | Transfer complete code. This 6-bit code is used to set the relevant bit in CER or IPR of the EDMA3PCC module. |
| 11 | Reserved | 0 | Reserved |
| 10-8 | FWID | 0-7h 0 1h 2h 3h 4h 5h 6h-7h | FIFO width. Applies if either SAM or DAM is set to FIFO mode. FIFO width is 8-bit. FIFO width is 16-bit. FIFO width is 32-bit. FIFO width is 64-bit. FIFO width is 128-bit. FIFO width is 256-bit. Reserved |
| 7 | Reserved | 0 | Reserved |
| 6-4 | PRI | 0-7h 0 1h-6h 7h | Transfer priority. Priority 0 - Highest priority Priority 1 to priority 6 Priority 7 - Lowest priority |
| 3-2 | Reserved | 0 | Reserved |
| 1 | DAM | 0 1 | Destination address mode within an array. INCR: Destination addressing within an array increments. FIFO: Destination addressing within an array wraps around upon reaching FIFO width. |

Table 4-85. Destination FIFO Options Register (DFOPT_n) Field Descriptions (continued)

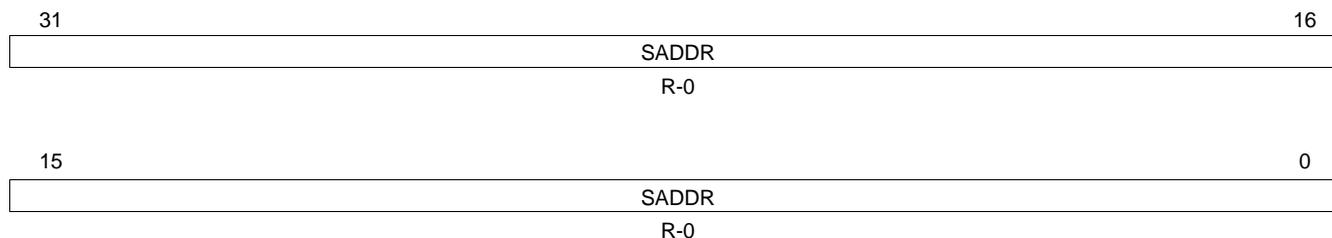
| Bit | Field | Value | Description |
|-----|-------|-------|--|
| 0 | SAM | | Source address mode within an array. |
| | | 0 | INCR: Source addressing within an array increments. |
| | | 1 | FIFO: Source addressing within an array wraps around upon reaching FIFO width. |

4.3.6.11 Destination FIFO Source Address Register (DFSRC n)

The destination FIFO source address register (DFSRC n) is shown in [Figure 4-84](#) and described in [Table 4-86](#).

Note: The value for n varies from 0 to DSTREGDEPTH for the given EDMA3TC.

Figure 4-84. Destination FIFO Source Address Register (DFSRC n)



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

Table 4-86. Destination FIFO Source Address Register (DFSRC n) Field Descriptions

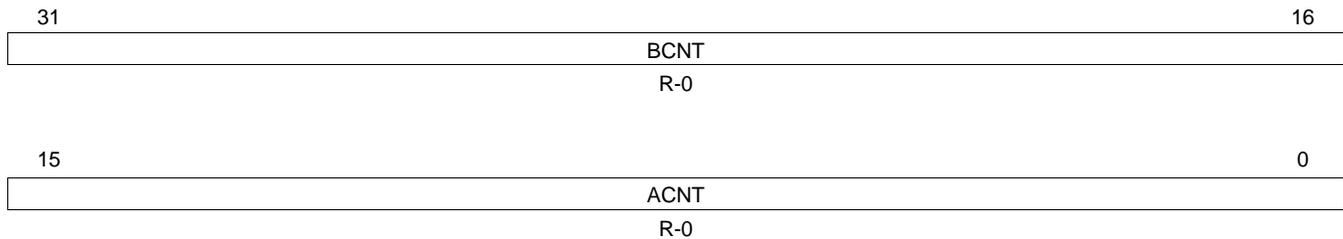
| Bit | Field | Value | Description |
|------|-------|-------|-------------------|
| 31-0 | SADDR | 0 | Always read as 0. |

4.3.6.12 Destination FIFO Count Register (DFCNT n)

The destination FIFO count register (DFCNT n) is shown in [Figure 4-85](#) and described in [Table 4-87](#).

Note: The value for n varies from 0 to DSTREGDEPTH for the given EDMA3TC.

Figure 4-85. Destination FIFO Count Register (DFCNT n)



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

Table 4-87. Destination FIFO Count Register (DFCNT n) Field Descriptions

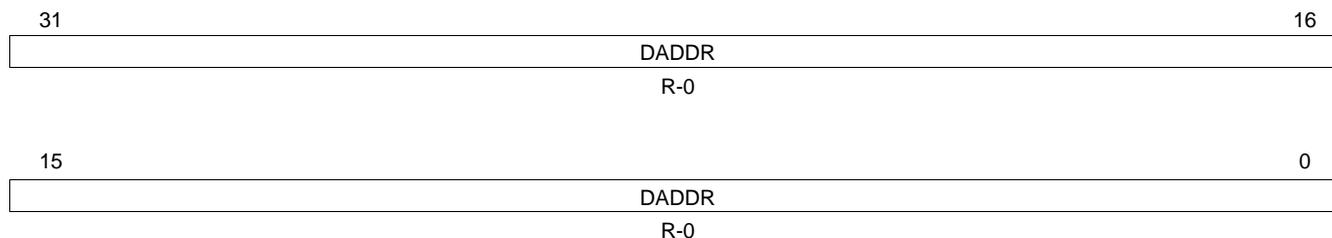
| Bit | Field | Value | Description |
|-------|-------|---------|---|
| 31-16 | BCNT | 0-FFFFh | B-dimension count. Number of arrays to be transferred, where each array is ACNT in length. Count/count remaining for destination register set. Represents the amount of data remaining to be written. |
| 15-0 | ACNT | 0-FFFFh | A-dimension count. Number of bytes to be transferred in first dimension count/count remaining for destination register set. Represents the amount of data remaining to be written. |

4.3.6.13 Destination FIFO Destination Address Register (DFDST n)

The destination FIFO destination address register (DFDST n) is shown in [Figure 4-86](#) and described in [Table 4-88](#).

Note: The value for n varies from 0 to DSTREGDEPTH for the given EDMA3TC.

Figure 4-86. Destination FIFO Destination Address Register (DFDST n)



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

Table 4-88. Destination FIFO Destination Address Register (DFDST n) Field Descriptions

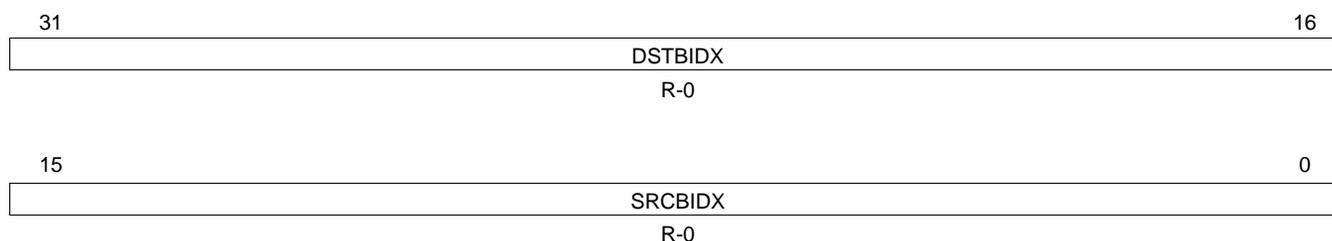
| Bit | Field | Value | Description |
|------|-------|-------|--|
| 31-0 | DADDR | 0 | Destination address for the destination FIFO register set. When a transfer request (TR) is complete, the final value should be the address of the last write command issued. |

4.3.6.14 Destination FIFO B-Index Register (DFBIDX n)

The destination FIFO B-index register (DFBIDX n) is shown in [Figure 4-87](#) and described in [Table 4-89](#).

Note: The value for n varies from 0 to DSTREGDEPTH for the given EDMA3TC.

Figure 4-87. Destination FIFO B-Index Register (DFBIDX n)



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

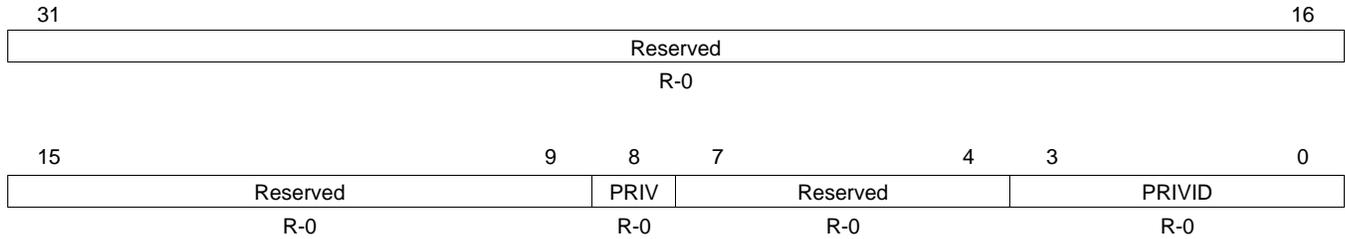
Table 4-89. Destination FIFO B-Index Register (DFBIDX n) Field Descriptions

| Bit | Field | Value | Description |
|-------|---------|---------|---|
| 31-16 | DSTBIDX | 0-FFFFh | B-Index offset between destination arrays. Represents the offset in bytes between the starting address of each destination. |
| 15-0 | SRCBIDX | 0 | Always read as 0. |

4.3.6.15 Destination FIFO Memory Protection Proxy Register (DFMPPRXY_n)

The destination FIFO memory protection proxy register (DFMPPRXY_n) is shown in [Figure 4-88](#) and described in [Table 4-81](#).

Figure 4-88. Destination FIFO Memory Protection Proxy Register (DFMPPRXY_n)



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

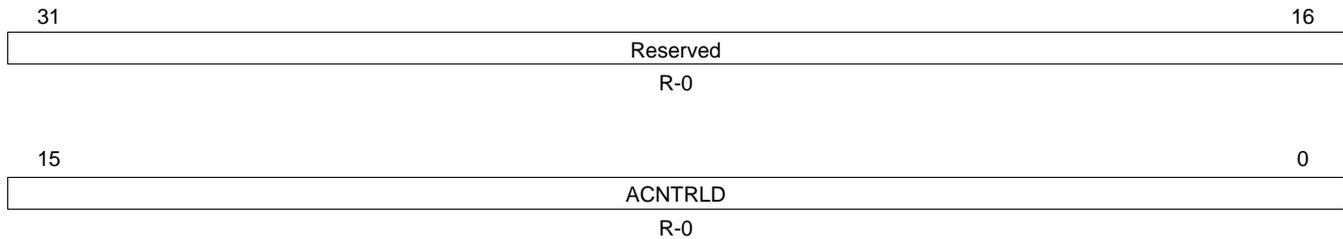
Table 4-90. Destination FIFO Memory Protection Proxy Register (DFMPPRXY_n) Field Descriptions

| Bit | Field | Value | Description |
|------|----------|--------|---|
| 31-9 | Reserved | 0 | Reserved |
| 8 | PRIV | 0 1 | Privilege level. This contains the Privilege level used by the EDMA3 programmer to set up the parameter entry in the channel controller. This field is set up when the associated TR is submitted to the EDMA3TC. The privilege ID is used while issuing read and write command to the target endpoints so that the target endpoints can perform memory protection checks based on the PRIV of the host that set up the DMA transaction. 0 User-level privilege 1 Supervisor-level privilege |
| 7-4 | Reserved | 0 | Reserved |
| 3-0 | PRIVID | 0-Fh | Privilege ID. This contains the Privilege ID of the EDMA3 programmer that set up the parameter entry in the channel controller. This field is set up when the associated TR is submitted to the EDMA3TC. This PRIVID value is used while issuing read and write commands to the target endpoints so that the target endpoints can perform memory protection checks based on the PRIVID of the host that set up the DMA transaction. |

4.3.6.16 Destination FIFO Count Reload Register (DFCNTRLD)

The destination FIFO count reload register (DFCNTRLD) is shown in [Figure 4-89](#) and described in [Table 4-91](#).

Figure 4-89. Destination FIFO Count Reload Register (DFCNTRLD)



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

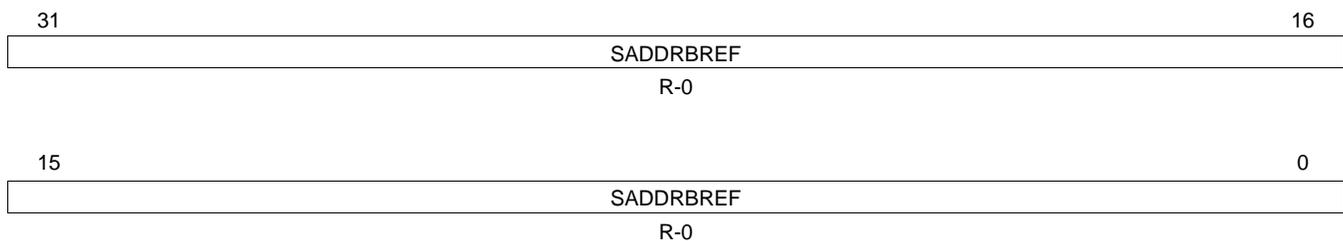
Table 4-91. Destination FIFO Count Reload Register (DFCNTRLD) Field Descriptions

| Bit | Field | Value | Description |
|-------|----------|---------|---|
| 31-16 | Reserved | 0 | Reserved |
| 15-0 | ACNTRLD | 0-FFFFh | A-count reload value. Represents the originally programmed value of ACNT. The reload value is used to reinitialize ACNT after each array is serviced. |

4.3.6.17 Destination FIFO Source Address B-Reference Register (DFSRCBREF)

The destination FIFO source address B-reference register (DFSRCBREF) is shown in [Figure 4-90](#) and described in [Table 4-92](#).

Figure 4-90. Destination FIFO Source Address B-Reference Register (DFSRCBREF)



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

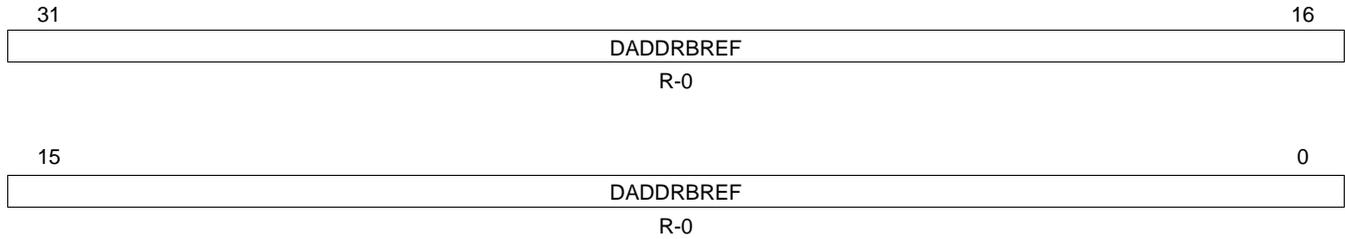
Table 4-92. Destination FIFO Source Address B-Reference Register (DFSRCBREF) Field Descriptions

| Bit | Field | Value | Description |
|------|-----------|-------|-----------------------------------|
| 31-0 | SADDRBREF | 0 | Not Applicable. Always read as 0. |

4.3.6.18 Destination FIFO Destination Address B-Reference (DFDSTBREF)

The destination FIFO destination address B-reference register (DFDSTBREF) is shown in [Figure 4-91](#) and described in [Table 4-93](#).

Figure 4-91. Destination FIFO Destination Address B-Reference Register (DFDSTBREF)



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

Table 4-93. Destination FIFO Destination Address B-Reference Register (DFDSTBREF) Field Descriptions

| Bit | Field | Value | Description |
|------|-----------|--------------|---|
| 31-0 | DADDRBREF | 0-FFFF FFFFh | Destination address reference for the destination FIFO register set. Represents the starting address for the array currently being written. |

A.1 Debug Checklist

This section lists some tips to keep in mind while debugging applications using the EDMA3.

The following provides some common issues and their probable causes and resolutions.

- The transfer associated with the channel does not happen. The channel does not get serviced. The EDMA3CC may not service a transfer request, even though the associated PaRAM set is programmed appropriately. Check for the following:
 1. Verify that events are enabled, i.e., if an external/peripheral event is latched in Event Registers (ER/ERH), make sure that the event is enabled in the Event Enable Registers (EER/EERH). Similarly, for QDMA channels, make sure that QDMA events are appropriately enabled in the QDMA Event Enable Register (QEER).
 2. Verify that the DMA or QDMA Secondary Event Register (SER/SERH/QSERH) 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:
 - QDMA channels: Typically if the parameter set is non-static and expected to be terminated by a NULL set (i.e., OPT.STATIC = 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 QEMR and QSER. This disables further prioritization of the channel.
 - DMA channels used in a continuous mode: The peripheral may be set up to continuously generate infinite events (for instance, in case of McBSP, 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 generates additional events, an error condition is set in the SER.Ex and EMR.Ex set, preventing further event prioritization. You must ensure that the number of events received is limited to the expected number of events for which the parameter set is programmed, or you must ensure that bits corresponding to particular channel or event are not set in the Secondary event registers (SER/SERH/QSER) and Event Missed Registers (EMR/EMRH/QEMR) before trying to perform subsequent transfers for the event/channel.
- Completion interrupts are not asserted, or no further interrupts are received after the first completion interrupt. You must ensure the following:
 1. The interrupts are enabled in the EDMA3 Channel Controller, via the Interrupt Enable Registers (IER/IERH).
 2. The corresponding interrupts are enabled in the device interrupt controller.
 3. The set interrupts are cleared in the interrupt pending registers (IPR/IPRH) before exiting the transfer completion interrupt service routine (ISR). For details on writing EDMA3 ISRs, see [Section 2.9.1.2](#).
 4. If working with shadow region interrupts, make sure that the DMA Region Access registers (DRAE/DRAEH) are set up properly, because the DRAE/DRAEH registers act as secondary enables for shadow region completion interrupts, along with the IER/IERH registers.

If working with shadow region interrupts, make sure that the bits corresponding to the transfer completion code (TCC) value are also enabled in the DRAE/DRAEH registers. For instance, if the PaRAM set associated with Channel 0 returns a completion code of 63 (OPT.TCC=63), ensure that DRAEH.E63 is also set for a shadow region completion interrupt because the interrupt pending register bit set is IPRH.I63 (not IPR.I0).

A.2 Miscellaneous Programming/Debug Tips

1. For several registers, the setting and clearing of bits needs to be done via separate dedicated registers. For example, the Event Register (ER/ERH) can only be cleared by writing a 1 to the corresponding bits in the Event Clear Registers (ECR/ECRH). Similarly, the Event Enable Register (EER/EERH) bits can only be set with writes of 1 to the Event Enable Set Registers (EESR/EESRH) and cleared with writes of 1 to the corresponding bits in the Event Enable Clear Register (EECR/EECRH).
2. Writes to the shadow region memory maps are governed by region access registers (DRAE/DRAEH/QRAE). If the appropriate channels are not enabled in these registers, read/write access to the shadow region memory map is not enabled.
3. When working with shadow region completion interrupts, ensure that the DMA Region Access Registers (DRAE/DRAEH) for every region are set in a mutually exclusive way (unless it is a requirement for an application). If there is an overlap in the allocated channels and transfer completion codes (setting of Interrupt Pending Register bits) in the region resource allocation, it results in multiple shadow region completion interrupts. For example, if DRAE0.E0 and DRAE1.E1 are both set, then on completion of a transfer that returns a TCC=0, they generate both shadow region 0 and 1 completion interrupts.
4. While programming a non-dummy parameter set, ensure the CCNT is not left to zero.
5. Enable the EDMA3CC error interrupt in the device controller and attach an interrupt service routine (ISR) to ensure that error conditions are not missed in an application and are appropriately addressed with the ISR.
6. Depending on the application, you may want to break large transfers into smaller transfers and use self-chaining to prevent starvation of other events in an event queue.
7. In applications where a large transfer is broken into sets of small transfers using chaining or other methods, you might choose to use the early chaining option to reduce the time between the sets of transfers and increase the throughput. However, keep in mind that with early completion, all data might have been received at the end point as soon as completion is reported because the EDMA3CC internally signals completion.
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).

Setting Up a Transfer

The following list provides a quick guide for the typical steps involved in setting up a transfer.

1. Initiating a DMA/QDMA channel
 - a. Determine the type of channel (QDMA or DMA) to be used.
 - b. Channel mapping
 - If using a QDMA Channel, program the QCHMAP with the parameter set number to which the channel maps and the trigger word.
 - If using a DMA channel, program the DCHMAP 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 DRAE/DRAEH 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 2.7.1](#).)
 - d. Determine the type of triggering used.
 - If external events are used for triggering (DMA channels), enable the respective event in EER/EERH by writing into EESR/EESRH.
 - If QDMA Channel is used, enable the channel in QEER by writing into QEESR.
 - e. Queue setup
 - If a QDMA channel is used, set up the QDMAQNUM to map the channel to the respective event queue.
 - If a DMA channel is used, set up the DMAQNUM to map the event to the respective event queue.
2. Parameter set setup.

Program the PaRAM set number associated with the channel. 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 above) just before the write to the trigger word.

For parameter set field setups for different types of transfers, see [Chapter 3](#). For details on how to set up final/intermediate completion chaining and/or interrupts, see [Section 2.8](#) and [Section 2.9](#).
3. Interrupt setup
 - a. Enable the interrupt in the IER/IERH by writing into IESR/IESRH.
 - b. Ensure that the EDMA3CC completion interrupt (either the global or the shadow region interrupt) is enabled properly in the device interrupt controller.
 - c. Set up the interrupt controller properly to receive the expected EDMA3 interrupt.
4. Initiate transfer.

This step is highly dependent on the event trigger source:

 - If the source is an external event coming from a peripheral, the peripheral is enabled to start generating relevant EDMA3 events that can be latched to the ER transfer.
 - For QDMA events, writes to the trigger word (step 2 above) initiate the transfer.
 - Manually triggered transfers are initiated by writes to the Event Set Registers (ESR/ESRH).
 - Chained-trigger events initiate when a previous transfer returns a transfer completion code equal to the chained channel number.

5. Wait for completion

- If the interrupts are enabled as mentioned in step 3 above, then the EDMA3CC generates a completion interrupt to the CPU whenever transfer completion results in setting the corresponding bits in the interrupt pending register (IPR\IPRH). The set bits must be cleared in the IPR\IPRH by writing to corresponding bit in ICR\ICRH.
- 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 IPR\IPRH. Again, the set bits in the IPR\IPRH must be manually cleared via ICR\ICRH before the next set of transfers is performed for the same transfer completion code values.

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All products are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by government requirements, testing of all parameters of each product is not necessarily performed.

TI assumes no liability for applications assistance or customer product design. Customers are responsible for their products and applications using TI components. To minimize the risks associated with customer products and applications, customers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any TI patent right, copyright, mask work right, or other TI intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information published by TI regarding third-party products or services does not constitute a license from TI to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of TI information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. Reproduction of this information with alteration is an unfair and deceptive business practice. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions.

Resale of TI products or services with statements different from or beyond the parameters stated by TI for that product or service voids all express and any implied warranties for the associated TI product or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

TI products are not authorized for use in safety-critical applications (such as life support) where a failure of the TI product would reasonably be expected to cause severe personal injury or death, unless officers of the parties have executed an agreement specifically governing such use. Buyers represent that they have all necessary expertise in the safety and regulatory ramifications of their applications, and acknowledge and agree that they are solely responsible for all legal, regulatory and safety-related requirements concerning their products and any use of TI products in such safety-critical applications, notwithstanding any applications-related information or support that may be provided by TI. Further, Buyers must fully indemnify TI and its representatives against any damages arising out of the use of TI products in such safety-critical applications.

TI products are neither designed nor intended for use in military/aerospace applications or environments unless the TI products are specifically designated by TI as military-grade or "enhanced plastic." Only products designated by TI as military-grade meet military specifications. Buyers acknowledge and agree that any such use of TI products which TI has not designated as military-grade is solely at the Buyer's risk, and that they are solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI products are neither designed nor intended for use in automotive applications or environments unless the specific TI products are designated by TI as compliant with ISO/TS 16949 requirements. Buyers acknowledge and agree that, if they use any non-designated products in automotive applications, TI will not be responsible for any failure to meet such requirements.

Following are URLs where you can obtain information on other Texas Instruments products and application solutions:

Products

| | |
|-----------------------------|--|
| Amplifiers | amplifier.ti.com |
| Data Converters | dataconverter.ti.com |
| DLP® Products | www.dlp.com |
| DSP | dsp.ti.com |
| Clocks and Timers | www.ti.com/clocks |
| Interface | interface.ti.com |
| Logic | logic.ti.com |
| Power Mgmt | power.ti.com |
| Microcontrollers | microcontroller.ti.com |
| RFID | www.ti-rfid.com |
| RF/IF and ZigBee® Solutions | www.ti.com/lprf |

Applications

| | |
|--------------------|--|
| Audio | www.ti.com/audio |
| Automotive | www.ti.com/automotive |
| Broadband | www.ti.com/broadband |
| Digital Control | www.ti.com/digitalcontrol |
| Medical | www.ti.com/medical |
| Military | www.ti.com/military |
| Optical Networking | www.ti.com/opticalnetwork |
| Security | www.ti.com/security |
| Telephony | www.ti.com/telephony |
| Video & Imaging | www.ti.com/video |
| Wireless | www.ti.com/wireless |

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265
Copyright © 2009, Texas Instruments Incorporated