

Intel® Xeon® Processor E7 Family Uncore Performance Monitoring Programming Guide

Reference Number: 325294-001

April 2011

INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH INTEL PRODUCTS. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. EXCEPT AS PROVIDED IN INTEL'S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, INTEL ASSUMES NO LIABILITY WHATSOEVER AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO SALE AND/OR USE OF INTEL PRODUCTS INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

UNLESS OTHERWISE AGREED IN WRITING BY INTEL, THE INTEL PRODUCTS ARE NOT DESIGNED NOR INTENDED FOR ANY APPLICATION IN WHICH THE FAILURE OF THE INTEL PRODUCT COULD CREATE A SITUATION WHERE PERSONAL INJURY OR DEATH MAY OCCUR.

Intel may make changes to specifications and product descriptions at any time, without notice. Designers must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined." Intel reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them. The information here is subject to change without notice. Do not finalize a design with this information.

The Intel® 64 architecture processors described in this document may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request. Contact your local Intel sales office or your distributor to obtain the latest specifications and before placing your product order.

Requires a system with a 64-bit enabled processor, chipset, BIOS and software. Performance will vary depending on the specific hardware and software you use. Consult your PC manufacturer for more information. For more information, visit <http://www.intel.com/info/em64t>

Intel and Intel Xeon are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

*Other names and brands may be claimed as the property of others.

Contact your local Intel sales office or your distributor to obtain the latest specifications and before placing your product order.

Copies of documents which have an order number and are referenced in this document, or other Intel literature, may be obtained by calling 1-800-548-4725, or go to: <http://www.intel.com/design/literature.htm>

Copyright © 2011 Intel Corporation

CONTENTS

CHAPTER 1 INTRODUCTION

| | | |
|-----|---------------------------|-----|
| 1.1 | INTRODUCTION | 1-1 |
| 1.2 | UNCORE PMU OVERVIEW | 1-1 |
| 1.3 | UNCORE PMU SUMMARY TABLES | 1-2 |
| 1.4 | REFERENCES | 1-3 |

CHAPTER 2 UNCORE PERFORMANCE MONITORING

| | | |
|---------|---|------|
| 2.1 | GLOBAL PERFORMANCE MONITORING CONTROL COUNTER OVERFLOW | 2-1 |
| 2.1.1 | Freezing on Counter Overflow | 2-1 |
| 2.1.2 | PMI on Counter Overflow | 2-1 |
| 2.1.3 | Setting up a Monitoring Session | 2-1 |
| 2.1.4 | Reading the Sample Interval | 2-2 |
| 2.1.5 | Enabling a New Sample Interval from Frozen Counters | 2-2 |
| 2.1.6 | Global Performance Monitors | 2-3 |
| 2.1.6.1 | Global PMON Global Control/Status Registers | 2-3 |
| 2.2 | U-BOX PERFORMANCE MONITORING | 2-5 |
| 2.2.1 | U-Box PMON Summary | 2-5 |
| 2.2.1.1 | U-Box Box Level PMON State | 2-5 |
| 2.2.1.2 | U-Box PMON state - Counter/Control Pairs | 2-5 |
| 2.2.2 | U-Box Performance Monitoring Events | 2-6 |
| 2.2.3 | U-Box Events Ordered By Code | 2-7 |
| 2.2.4 | U-Box Performance Monitor Event List | 2-7 |
| 2.3 | C-BOX PERFORMANCE MONITORING | 2-9 |
| 2.3.1 | Overview of the C-Box | 2-9 |
| 2.3.2 | C-Box Performance Monitoring Overview | 2-10 |
| 2.3.2.1 | C-Box PMU - Overflow, Freeze and Unfreeze | 2-10 |
| 2.3.3 | C-BOX Performance Monitors | 2-11 |
| 2.3.3.1 | C-Box Box Level PMON state | 2-15 |
| 2.3.3.2 | C-Box PMON state - Counter/Control Pairs | 2-15 |
| 2.3.4 | C-BOX Performance Monitoring Events | 2-17 |
| 2.3.4.1 | An Overview | 2-17 |
| 2.3.4.2 | Acronyms frequently used in C-Box Events | 2-17 |
| 2.3.4.3 | The Queues | 2-18 |
| 2.3.4.4 | Detecting Performance Problems in the C-Box Pipeline | 2-18 |
| 2.3.5 | C-Box Events Ordered By Code | 2-19 |
| 2.3.6 | C-Box Performance Monitor Event List | 2-20 |
| 2.4 | B-BOX PERFORMANCE MONITORING | 2-33 |
| 2.4.1 | Overview of the B-Box | 2-33 |
| 2.4.2 | B-Box Performance Monitoring Overview | 2-33 |
| 2.4.2.1 | B-Box PMU - On Overflow and the Consequences (PMI/Freeze) | 2-34 |
| 2.4.3 | B-BOX Performance Monitors | 2-34 |
| 2.4.3.1 | B-Box Box Level PMON state | 2-35 |
| 2.4.3.2 | B-Box PMON state - Counter/Control Pairs + Filters | 2-36 |
| 2.4.4 | B-Box Performance Monitoring Events | 2-37 |
| 2.4.4.1 | On the ARBQ | 2-38 |
| 2.4.4.2 | On the Major B-Box Structures | 2-38 |
| 2.4.4.3 | On InvToE Transactions | 2-38 |
| 2.4.5 | B-Box Events Ordered By Code | 2-39 |
| 2.4.6 | B-Box Performance Monitor Event List | 2-40 |
| 2.5 | S-BOX PERFORMANCE MONITORING | 2-48 |
| 2.5.1 | Overview of the S-Box | 2-48 |
| 2.5.2 | S-Box Performance Monitoring Overview | 2-48 |
| 2.5.2.1 | S-Box PMU - Overflow, Freeze and Unfreeze | 2-48 |
| 2.5.3 | S-BOX Performance Monitors | 2-49 |
| 2.5.3.1 | S-Box PMON for Global State | 2-50 |
| 2.5.3.2 | S-Box Box Level PMON state | 2-50 |
| 2.5.3.3 | S-Box PMON state - Counter/Control Pairs + Filters | 2-51 |
| 2.5.3.4 | S-Box Registers for Mask/Match Facility | 2-52 |
| 2.5.4 | S-BOX Performance Monitoring Events | 2-54 |
| 2.5.4.1 | An Overview | 2-54 |
| 2.5.4.2 | On Queue Occupancy Usage | 2-54 |

| | | |
|---------|---|-------|
| 2.5.4.3 | On Packet Transmission Events | 2-55 |
| 2.5.5 | S-Box Events Ordered By Code | 2-56 |
| 2.5.6 | S-Box Performance Monitor Event List | 2-58 |
| 2.6 | R-BOX PERFORMANCE MONITORING | 2-78 |
| 2.6.1 | Overview of the R-Box | 2-78 |
| 2.6.1.1 | R-Box Input Port | 2-78 |
| 2.6.1.2 | R-Box Arbitration Control | 2-79 |
| 2.6.1.3 | R-Box Output Port | 2-79 |
| 2.6.1.4 | R-Box Link Layer Resources | 2-79 |
| 2.6.2 | R-Box Performance Monitoring Overview | 2-79 |
| 2.6.2.1 | Choosing An Event To Monitor - Example | 2-80 |
| 2.6.2.2 | R-Box PMU - Overflow, Freeze and Unfreeze | 2-80 |
| 2.6.3 | R-BOX Performance Monitors | 2-80 |
| 2.6.3.1 | R-Box Performance Monitors To Port Mapping | 2-84 |
| 2.6.3.2 | R-Box Box Level PMON state | 2-84 |
| 2.6.3.3 | R-Box PMON state - Counter/Control Pairs + Filters | 2-85 |
| 2.6.3.4 | R-Box IPERF Performance Monitoring Control Registers | 2-88 |
| 2.6.3.5 | R-Box QLX Performance Monitoring Control Registers | 2-89 |
| 2.6.3.6 | R-Box Registers for Mask/Match Facility | 2-90 |
| 2.6.4 | R-BOX Performance Monitoring Events | 2-94 |
| 2.6.4.1 | An Overview: | 2-94 |
| 2.6.5 | R-Box Events Ordered By Code | 2-94 |
| 2.6.6 | R-Box Performance Monitor Event List | 2-95 |
| 2.7 | M-BOX PERFORMANCE MONITORING | 2-102 |
| 2.7.1 | Overview of the M-Box | 2-102 |
| 2.7.2 | Functional Overview | 2-102 |
| 2.7.2.1 | Intel® Xeon® Processor E7 Family Scalable Memory Buffer | 2-103 |
| 2.7.3 | M-Box Performance Monitoring Overview | 2-104 |
| 2.7.3.1 | Choosing An Event To Monitor - Example using subcontrol registers | 2-104 |
| 2.7.3.2 | M-Box PMU - Overflow, Freeze and Unfreeze | 2-105 |
| 2.7.4 | M-BOX Performance Monitors | 2-106 |
| 2.7.4.1 | M-Box Box Level PMON state | 2-107 |
| 2.7.4.2 | M-Box PMON state - Counter/Control Pairs | 2-108 |
| 2.7.4.3 | M-Box PMU Filter Registers | 2-110 |
| 2.7.4.4 | M-Box PMU Subcontrol Registers - Subunit descriptions | 2-110 |
| 2.7.5 | M-Box Performance Monitoring Events | 2-118 |
| 2.7.5.1 | An Overview: | 2-118 |
| 2.7.6 | M-Box Events Ordered By Code | 2-119 |
| 2.7.7 | M-Box Performance Monitor Event List | 2-120 |
| 2.8 | W-BOX PERFORMANCE MONITORING | 2-134 |
| 2.8.1 | Overview of the W-Box | 2-134 |
| 2.8.2 | W-Box Performance Monitoring Overview | 2-134 |
| 2.8.2.1 | W-Box PMU - Overflow, Freeze and Unfreeze | 2-134 |
| 2.8.3 | W-BOX Performance Monitors | 2-135 |
| 2.8.3.1 | W-Box Box Level PMON state | 2-135 |
| 2.8.3.2 | W-Box PMON state - Counter/Control Pairs | 2-136 |
| 2.8.4 | W-BOX Performance Monitoring Events | 2-138 |
| 2.8.4.1 | An Overview: | 2-138 |
| 2.8.5 | W-Box Events Ordered By Code | 2-138 |
| 2.8.6 | W-Box Performance Monitor Event List | 2-139 |
| 2.9 | PACKET MATCHING REFERENCE | 2-140 |

FIGURES

| | | |
|-------------|---|-------|
| Figure 1-1. | Intel® Xeon® Processor E7 Family Block Diagram..... | 1-1 |
| Figure 2-1. | R-Box Block Diagram..... | 2-78 |
| Figure 2-2. | Memory Controller Block Diagram..... | 2-103 |

This page intentionally left blank

TABLES

| | | |
|-------------|--|-------|
| Table 1-1. | Per-Box Performance Monitoring Capabilities | 1-2 |
| Table 1-2. | Uncore Performance Monitoring MSRs | 1-2 |
| Table 2-1. | Global Performance Monitoring Control MSRs | 2-3 |
| Table 2-2. | U_MSR_PMON_GLOBAL_CTL Register – Field Definitions | 2-4 |
| Table 2-3. | U_MSR_PMON_GLOBAL_STATUS Register – Field Definitions | 2-4 |
| Table 2-4. | U_MSR_PMON_GLOBAL_OVF_CTL Register – Field Definitions | 2-5 |
| Table 2-5. | U-Box Performance Monitoring MSRs | 2-5 |
| Table 2-6. | U_MSR_PMON_EVT_SEL Register – Field Definitions | 2-6 |
| Table 2-7. | U_MSR_PMON_CTR Register – Field Definitions | 2-6 |
| Table 2-8. | Performance Monitor Events for U-Box Events | 2-7 |
| Table 2-9. | C-Box Performance Monitoring MSRs | 2-11 |
| Table 2-10. | C_MSR_PMON_GLOBAL_CTL Register – Field Definitions | 2-15 |
| Table 2-11. | C_MSR_PMON_GLOBAL_STATUS Register – Field Definitions | 2-15 |
| Table 2-12. | C_MSR_PMON_GLOBAL_OVF_CTL Register – Field Definitions | 2-15 |
| Table 2-13. | C_MSR_PMON_EVT_SEL{5-0} Register – Field Definitions | 2-16 |
| Table 2-14. | C_MSR_PMON_CTR{5-0} Register – Field Definitions | 2-17 |
| Table 2-15. | Performance Monitor Events for C-Box Events | 2-19 |
| Table 2-16. | B-Box Performance Monitoring MSRs | 2-34 |
| Table 2-17. | B_MSR_PMON_GLOBAL_CTL Register – Field Definitions | 2-35 |
| Table 2-18. | B_MSR_PMON_GLOBAL_STATUS Register – Field Definitions | 2-35 |
| Table 2-19. | B_MSR_PMON_GLOBAL_OVF_CTL Register – Field Definitions | 2-35 |
| Table 2-20. | B_MSR_PMON_EVT_SEL{3-0} Register – Field Definitions | 2-36 |
| Table 2-21. | B_MSR_PMON_CNT{3-0} Register – Field Definitions | 2-36 |
| Table 2-22. | B_MSR_MATCH_REG Register – Field Definitions | 2-37 |
| Table 2-23. | B_MSR_MASK_REG Register – Field Definitions | 2-37 |
| Table 2-24. | Performance Monitor Events for B-Box Events | 2-39 |
| Table 2-25. | S-Box Performance Monitoring MSRs | 2-49 |
| Table 2-26. | S_MSR_PMON_SUMMARY Register Fields | 2-50 |
| Table 2-27. | S_MSR_PMON_GLOBAL_CTL Register Fields | 2-51 |
| Table 2-28. | S_MSR_PMON_GLOBAL_STATUS Register Fields | 2-51 |
| Table 2-29. | S_MSR_PMON_OVF_CTRL Register Fields | 2-51 |
| Table 2-30. | S_CSR_PMON_CTL{3-0} Register – Field Definitions | 2-52 |
| Table 2-31. | S_CSR_PMON_CTR{3-0} Register – Field Definitions | 2-52 |
| Table 2-32. | S_MSR_MM_CFG Register – Field Definitions | 2-53 |
| Table 2-33. | S_MSR_MATCH Register – Field Definitions | 2-53 |
| Table 2-34. | S_MSR_MATCH.opc - Opcode Match by Message Class | 2-54 |
| Table 2-35. | S_MSR_MASK Register – Field Definitions | 2-54 |
| Table 2-36. | S-Box Data Structure Occupancy Events | 2-55 |
| Table 2-37. | Performance Monitor Events for S-Box Events | 2-56 |
| Table 2-38. | Input Buffering Per Port | 2-79 |
| Table 2-39. | R-Box Performance Monitoring MSRs | 2-80 |
| Table 2-40. | R-Box Port Map | 2-84 |
| Table 2-41. | R_MSR_PMON_GLOBAL_CTL_{15_8, 7_0} Register Fields | 2-84 |
| Table 2-42. | R_MSR_PMON_GLOBAL_STATUS_{15_8, 7_0} Register Fields | 2-84 |
| Table 2-43. | R_MSR_PMON_OVF_CTL_{15_8, 7_0} Register Fields | 2-85 |
| Table 2-44. | R_MSR_PMON_CTL{15-0} Register – Field Definitions | 2-85 |
| Table 2-45. | R_MSR_PMON_CTL{15-8} Event Select | 2-86 |
| Table 2-46. | R_MSR_PMON_CTL{7-0} Event Select | 2-87 |
| Table 2-47. | R_MSR_PMON_CTR{15-0} Register – Field Definitions | 2-88 |
| Table 2-48. | R_MSR_PORT{7-0}_JPERF_CFG{1-0} Registers | 2-88 |
| Table 2-49. | R_MSR_PORT{7-0}_QLX_CFG Register Fields | 2-89 |
| Table 2-50. | R_MSR_PORT{7-0}_XBR_SET{2-1}_MM_CFG Registers | 2-91 |
| Table 2-51. | R_MSR_PORT{7-0}_XBR_SET{2-1}_MATCH Registers | 2-92 |
| Table 2-52. | R_MSR_PORT{7-0}_XBR_SET{2-1}_MASK Registers | 2-93 |
| Table 2-53. | Message Events Derived from the Match/Mask filters | 2-93 |
| Table 2-54. | Performance Monitor Events for R-Box Events | 2-94 |
| Table 2-55. | Unit Masks for ALLOC_TO_ARB | 2-95 |
| Table 2-56. | Unit Masks for EOT_DEPTH_ACC | 2-96 |
| Table 2-57. | Unit Masks for EOT_ROLL_DEPTH_ACC | 2-96 |
| Table 2-58. | Unit Masks for GLOBAL_ARB_BID_FAIL | 2-98 |
| Table 2-59. | Unit Masks for NEW_PACKETS_RECV | 2-99 |
| Table 2-60. | Unit Masks for QUE_ARB_BID | 2-100 |

| | | |
|--------------|---|-------|
| Table 2-61. | Unit Masks for QUE_ARB_BID_FAIL | 2-101 |
| Table 2-62. | Unit Masks for TARGET_AVAILABLE | 2-102 |
| Table 2-63. | M-Box Performance Monitoring MSRs | 2-106 |
| Table 2-64. | M_MSR_PERF_GLOBAL_CTL Register Fields | 2-108 |
| Table 2-65. | M_MSR_PERF_GLOBAL_STATUS Register Fields | 2-108 |
| Table 2-66. | M_MSR_PERF_GLOBAL_OVF_CTL Register Fields | 2-108 |
| Table 2-67. | M_MSR_PMU_CNT_CTL{5-0} Register – Field Definitions | 2-109 |
| Table 2-68. | M_MSR_PMU_CNT_{5-0} Register – Field Definitions | 2-109 |
| Table 2-69. | M_MSR_PMU_TIMESTAMP_UNIT Register – Field Definitions | 2-110 |
| Table 2-70. | M_MSR_PMU_MM_CFG Register – Field Definitions | 2-110 |
| Table 2-71. | M_MSR_PMU_ADDR_MATCH Register – Field Definitions | 2-110 |
| Table 2-72. | M_MSR_PMU_ADDR_MASK Register – Field Definitions | 2-110 |
| Table 2-73. | M_MSR_PMU_DSP Register – Field Definitions | 2-112 |
| Table 2-74. | M_CSR_ISS_PMU Register – Field Definitions | 2-112 |
| Table 2-75. | M_MSR_PMU_MAP Register – Field Definitions | 2-113 |
| Table 2-76. | M_CSR_PMU_MA_MSC_THR Register – Field Definitions | 2-113 |
| Table 2-77. | TRP_PT_{DN,UP}_CND Encodings | 2-114 |
| Table 2-78. | M_MSR_PMU_PGT Register – Field Definitions | 2-114 |
| Table 2-79. | M_MSR_PMU_PLD Register – Field Definitions | 2-115 |
| Table 2-80. | M_MSR_PMU_ZDP_CTL_FVC Register – Field Definitions | 2-116 |
| Table 2-81. | M_MSR_PMU_ZDP_CTL_FVC.evnt{4-1} Encodings | 2-117 |
| Table 2-82. | M_MSR_PMU_ZDP_CTL_FVC.RESP Encodings | 2-117 |
| Table 2-83. | M_MSR_PMU_ZDP_CTL_FVC.BCMD Encodings | 2-118 |
| Table 2-84. | Performance Monitor Events for M-Box Events | 2-119 |
| Table 2-85. | Unit Masks for BCMD_SCHEDQ_OCCUPANCY | 2-120 |
| Table 2-86. | Unit Masks for CYCLES_DSP_FILL | 2-121 |
| Table 2-87. | Unit Masks for CYCLES_PGT_STATE | 2-121 |
| Table 2-88. | Unit Masks for CYCLES_SCHED_MODE | 2-122 |
| Table 2-89. | Unit Masks for FVC_EVO | 2-126 |
| Table 2-90. | Unit Masks for FVC_EV1 | 2-127 |
| Table 2-91. | Unit Masks for FVC_EV2 | 2-128 |
| Table 2-92. | Unit Masks for FVC_EV3 | 2-129 |
| Table 2-93. | W-Box Performance Monitoring MSRs | 2-135 |
| Table 2-94. | W_MSR_PMON_GLOBAL_CTL Register Fields | 2-135 |
| Table 2-95. | W_MSR_PMON_GLOBAL_STATUS Register Fields | 2-136 |
| Table 2-96. | W_MSR_PMON_GLOBAL_OVF_CTRL Register Fields | 2-136 |
| Table 2-97. | W_MSR_PMON_EVT_SEL_{3-0} Register – Field Definitions | 2-137 |
| Table 2-98. | W_MSR_PMON_FIXED_CTR_CTL Register – Field Definitions | 2-137 |
| Table 2-99. | W_MSR_PMON_CTR_{3-0} Register – Field Definitions | 2-138 |
| Table 2-100. | W_MSR_PMON_FIXED_CTR Register – Field Definitions | 2-138 |
| Table 2-101. | Performance Monitor Events for W-Box Events | 2-138 |
| Table 2-102. | Intel® QuickPath Interconnect Packet Message Classes | 2-140 |
| Table 2-103. | Opcode Match by Message Class | 2-141 |
| Table 2-104. | Opcodes (Alphabetical Listing) | 2-142 |

CHAPTER 1 INTRODUCTION

1.1 INTRODUCTION

Figure 1-1 provides an Intel® Xeon® Processor E7 Family block diagram.

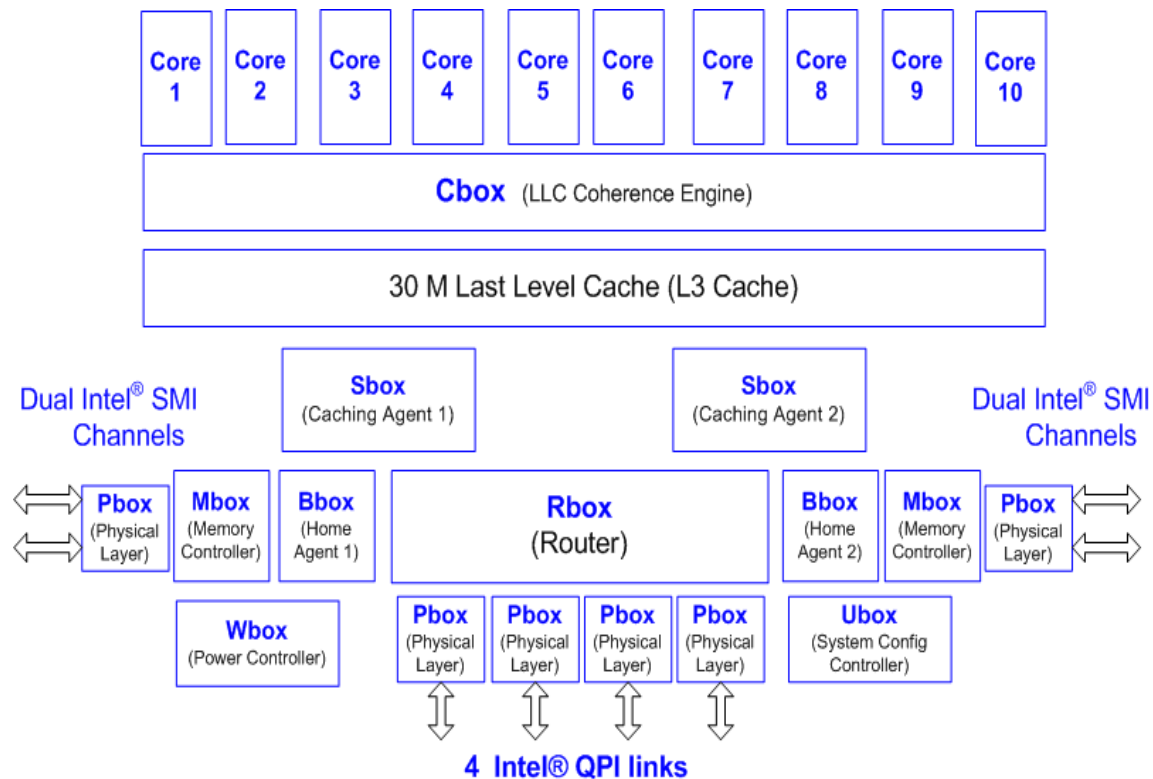


Figure 1-1. Intel® Xeon® Processor E7 Family Block Diagram

1.2 Uncore PMU Overview

The processor uncore performance monitoring is supported by PMUs local to each of the C, S, B, M, R, U, and W-Boxes. Each of these boxes communicates with the U-Box which contains registers to control all uncore PMU activity (as outlined in [Section 2.1, “Global Performance Monitoring Control Counter Overflow”](#)).

All processor uncore performance monitoring features can be accessed through RDMSR/WRMSR instructions executed at ring 0.

Since the uncore performance monitors represent socket-wide resources that are not context switched by the OS, it is highly recommended that only one piece of software (per-socket) attempt to program and extract information from the monitors. To keep things simple, it is also recommended that the monitoring software communicate with the OS such that it can be executed on coreId = 0, threadId = 0. Although recommended, this step is not necessary. Software may be notified of an overflowing uncore counter on any core.

The general performance monitoring capabilities in each box are outlined in the following table.

Table 1-1. Per-Box Performance Monitoring Capabilities

| Box | # Boxes | # Counters/ Box | Generic Counters? | Packet Match/ Mask Filters? | Bit Width |
|-------|---------------|-----------------------------------|----------------------|--------------------------------|-----------|
| C-Box | 10 | 6 | Y | N | 48 |
| S-Box | 2 | 6 | Y | Y | 48 |
| B-Box | 2 | 4 | N | Y | 48 |
| M-Box | 2 | 6 | N | N | 48 |
| R-Box | 1 (L/R sides) | 16 (2 per port, 8 per side) | N | Y | 48 |
| U-Box | 1 | 1 | Y | N | 48 |
| W-Box | 1 | 4 | Y | N | 48 |

1.3 Uncore PMU Summary Tables

Table 1-2. Uncore Performance Monitoring MSRs

| Box | MSR Addresses | Description |
|-----------------------|---------------|---------------------------------------|
| R-Box Counters | | |
| R-Box R | 0xE3F-0xE30 | Counter/Config Registers(15-8) |
| | 0xE2F-0xE2C | QLX SubConfig Registers for Ports 7-4 |
| | 0xE2B-0xE24 | IPERF 1 SubConfig Registers |
| | 0xE22-0xE20 | Global (Control/Status/Ovf Control) |
| R-Box L | 0xE1F-0xE10 | Counter/Config Registers(7-0) |
| | 0xE0F-0xE0C | QLX SubConfig Registers for Ports 3-0 |
| | 0xE0B-0xE04 | IPERF 0 SubConfig Registers |
| | 0xE02-0xE00 | Global (Control/Status/Ovf Control) |
| C-Box Counters | | |
| C-Box 9 | 0xFDB-0xFD0 | Counter/Config Registers |
| | 0xFC2-0xFC0 | Global (Control/Status/Ovf Control) |
| C-Box 8 | 0xF5B-0xF50 | Counter/Config Registers |
| | 0xF42-0xF40 | Global (Control/Status/Ovf Control) |
| C-Box 7 | 0xDFB-0xDF0 | Counter/Config Registers |
| | 0xDE2-0xDE0 | Global (Control/Status/Ovf Control) |
| C-Box 3 | 0xDDB-0xDD0 | Counter/Config Registers |
| | 0xDC2-0xDC0 | Global (Control/Status/Ovf Control) |
| C-Box 5 | 0xDBB-0xDB0 | Counter/Config Registers |
| | 0xDA2-0xDA0 | Global (Control/Status/Ovf Control) |
| C-Box 1 | 0xD9B-0xD90 | Counter/Config Registers |
| | 0xD82-0xDE0 | Global (Control/Status/Ovf Control) |
| C-Box 6 | 0xD7B-0xD70 | Counter/Config Registers |
| | 0xD62-0xD60 | Global (Control/Status/Ovf Control) |
| C-Box 2 | 0xD5B-0xD50 | Counter/Config Registers |

Table 1-2. Uncore Performance Monitoring MSRs

| Box | MSR Addresses | Description |
|-----------------------|---------------|---|
| | 0xD42-0xD40 | Global (Control/Status/Ovf Control) |
| C-Box 4 | 0xD3B-0xD30 | Counter/Config Registers |
| | 0xD22-0xD20 | Global (Control/Status/Ovf Control) |
| C-Box 0 | 0xD1B-0xD10 | Counter/Config Registers |
| | 0xD02-0xD00 | Global (Control/Status/Ovf Control) |
| M-Box Counters | | |
| M-Box 1 | 0xE5E-0xE5C | Match/Mask Registers |
| | 0xCFB-0xCF0 | Counter/Config Registers |
| | 0xCEB-0xCE4 | Subconfig Registers (FVC,PLD,PGT,THR,MAP,ISS,DSP) + Timestamp Register |
| | 0xCE2-0xCE0 | Global (Control/Status/Ovf Control) |
| M-Box 0 | 0xE56-0xE54 | Match/Mask Registers |
| | 0xCBB-0xCB0 | Counter/Config Registers |
| | 0xCAB-0xCA4 | Subconfig Registers (FVC,PLD,PGT,THR,MAP,ISS,DSP) + Timestamp Register |
| | 0xCA2-0xCA0 | Global (Control/Status/Ovf Control) |
| S-Box Counters | | |
| S-Box 1 | 0xE5A-0xE58 | Match/Mask Registers |
| | 0xCD7-0xCD0 | Counter/Config Registers |
| | 0xCC3-0xCC0 | Global (Control/Status/Ovf Control) |
| S-Box 0 | 0xE4A-0xE48 | Match/Mask Registers |
| | 0xC57-0xC50 | Counter/Config Registers |
| | 0xC43-0xC40 | Global (Control/Status/Ovf Control) |
| B-Box Counters | | |
| B-Box 1 | 0xE4E-0xE4D | Match/Mask Registers |
| | 0xC77-0xC70 | Counter/Config Registers |
| | 0xC62-0xC60 | Global (Control/Status/Ovf Control) |
| B-Box 0 | 0xE46-0xE45 | Match/Mask Registers |
| | 0xC37-0xC30 | Counter/Config Registers |
| | 0xC22-0xC20 | Global (Control/Status/Ovf Control) |
| U-Box Counters | | |
| U-Box | 0xC11-0xC10 | Counter/Config Registers |
| | 0xC02-0xC00 | Global (Control/Status/Ovf Control) |
| W-Box Counters | | |
| W-Box | 0x395-0x394 | Fixed Counter/Config Registers |
| | 0xC97-0xC90 | Counter/Config Registers |
| | 0xC82-0xC80 | Global (Control/Status/Ovf Control) |

1.4 References

The following sections provide a breakdown of the performance monitoring capabilities of each box.

- Section 2.1, “Global Performance Monitoring Control Counter Overflow”
- Section 2.2, “U-Box Performance Monitoring”
- Section 2.3, “C-Box Performance Monitoring”
- Section 2.4, “B-Box Performance Monitoring”
- Section 2.5, “S-Box Performance Monitoring”
- Section 2.6, “R-Box Performance Monitoring”
- Section 2.7, “M-Box Performance Monitoring”
- Section 2.8, “W-Box Performance Monitoring”
- Section 2.9, “Packet Matching Reference”

CHAPTER 2

UNCORE PERFORMANCE MONITORING

2.1 Global Performance Monitoring Control Counter Overflow

If a counter overflows, it will send the overflow signal towards the U-Box. This signal will be accumulated along the way in summary registers contained in each S-Box and a final summary register in the U-Box.

The Intel® Xeon® Processor E7 Family uncore performance monitors may be configured to respond to this overflow with two basic actions:

2.1.1 Freezing on Counter Overflow

Each uncore performance counter may be configured to, upon detection of overflow, disable (or 'freeze') all other counters in the uncore. To do so, the `.pmi_en` in the individual counter's control register must be set to 1. If the `U_MSR_PMON_GLOBAL_CTL.frz_all` is also set to 1, once the U-Box receives the PMI from the uncore box, it will set `U_MSR_PMON_GLOBAL_CTL.en_all` to 0 which will disable all counting.

2.1.2 PMI on Counter Overflow

The uncore may also be configured to, upon detection of a performance counter overflow, send a PMI signal to the core executing the monitoring software. To do so, the `.pmi_en` in the individual counter's control register must be set to 1 and `U_MSR_PMON_GLOBAL_CTL.pmi_core_sel` must be set to point to the core the monitoring software is executing on.

Note: PMI is decoupled from freeze, so if software also wants the counters frozen, it must set `U_MSR_PMON_GLOBAL_CTL.frz_all` to 1.

2.1.3 Setting up a Monitoring Session

On HW reset, all the counters should be disabled. Enabling is hierarchical. So the following steps must be taken to set up a new monitoring session:

a) Reset counters to ensure no stale values have been acquired from previous sessions:

- set `U_MSR_PMON_GLOBAL_CTL.rst_all` to 1.

b) Select event to monitor:

Determine what events should be captured and program the control registers to capture them (i.e. typically selected by programming the `.ev_sel/` bits although other bit fields may be involved).

i.e. Set `B_MSR_PMON_EVT_SEL3.ev_sel` to 0x03 to capture `SNP_MERGE`.

c) Enable counting locally:

i.e. Set `B_MSR_PMON_EVT_SEL3.en` to 1.

d) Enable counting at the box-level:

Enable counters within that box via it's 'GLOBAL_CTL' register

i.e. set B_MSR_PMON_GLOBAL_CTL[3] to 1.

e) Select how to gather data. If polling, skip to 4. If sampling:

To set up a **sample interval**, software can pre-program the data register with a value of [2^{48} - sample interval length]. Doing so allows software, through use of the pmi mechanism, to be **notified** when the number of events in the sample have been captured. Capturing a performance monitoring sample every 'X cycles' (the fixed counter in the W-Box counts uncore clock cycles) is a common use of this mechanism.

i.e. To stop counting and receive notification when the 1,000th SNP_MERGE has been detected,

- set B_MSR_PMON_CNT to (2^{48} - 1000)

- set B_MSR_PMON_EVT_SEL.pmi_en to 1

- set U_MSR_PMON_GLOBAL_CTL.frz_all to 1

- set U_MSR_PMON_GLOBAL_CTL.pmi_core_sel to which core the monitoring thread is executing on.

f) Enable counting at the global level by setting the U_MSR_PMON_GLOBAL_CTL.en_all bit to 1. Set the .rst_all field to 0 with the same write.

And with that, counting will begin.

2.1.4 Reading the Sample Interval

Software can either **poll** the counters whenever it chooses, or wait to be **notified** that a counter has overflowed (by receiving a PMI).

Polling - before reading, it is recommended that software freeze and disable the counters (by clearing U_MSR_PMON_GLOBAL_CTL.en_all).

Overflow bits are stored hierarchically within the Intel® Xeon® Processor E7 Family uncore. First, software should read the U_MSR_PMON_GLOBAL_STATUS.ov_* bits to determine whether a U or W box counter caused the overflow or whether it was a counter in a box attached to the S0 or S1 Box.

The S-Boxes aggregate overflow bits from the M/B/C/R boxes they are attached to. So the next step is to read the S{0,1}_MSR_PMON_SUMMARY.ov_* bits. Once the box(es) that contains the overflowing counter is identified, the last step is to read that box's *_MSR_PMON_GLOBAL_STATUS.ov field to find the overflowing counter.

Note: More than one counter may overflow at any given time.

2.1.5 Enabling a New Sample Interval from Frozen Counters

Note: Software can determine if the counters have been frozen due to a PMI by examining two bits: U_MSR_PMON_GLOBAL_SUMMARY.pmi should be 1 and U_MSR_PMON_GLOBAL_CTL.en_all should be 0. If not, set U_MSR_PMON_GLOBAL_CTL.en_all to 0 to disable counting.

a) Clear all uncore counters: Set U_MSR_PMON_GLOBAL_CTL.rst_all to 1.

b) Clear all overflow bits. When an overflow bit is cleared, all bits that summarize that overflow (above in the hierarchy) will also be cleared. Therefore it is only necessary to clear the overflow bits corresponding to the actual counter.

i.e. If counter 3 in B-Box 1 overflowed, to clear the overflow bit software should set `B_MSR_PMON_GLOBAL_OVF_CTL.clr_ov[3]` to 1 in B-Box 1. This action will also clear `S_MSR_PMON_SUMMARY.ov_mb` in S-Box 1 and `U_MSR_PMON_GLOBAL_STATUS.ov_s1.c`

c) Create the next sample: Reinitialize the sample by setting the monitoring data register to $(2^{48} - \text{sample_interval})$. Or set up a new sample interval as outlined in [Section 2.1.3, "Setting up a Monitoring Session"](#).

d) Re-enable counting: Set `U_MSR_PMON_GLOBAL_CTL.en_all` to 1. Set the `.rst_all` field back to 0 with the same write.

2.1.6 Global Performance Monitors

Table 2-1. Global Performance Monitoring Control MSRs

| MSR Name | Access | MSR Address | Size (bits) | Description |
|--|--------|-------------|-------------|------------------------------------|
| <code>U_MSR_PMON_GLOBAL_OVF_CTL</code> | RW_RW | 0x0C02 | 32 | U-Box PMON Global Overflow Control |
| <code>U_MSR_PMON_GLOBAL_STATUS</code> | RW_RO | 0x0C01 | 32 | U-Box PMON Global Status |
| <code>U_MSR_PMON_GLOBAL_CTL</code> | RW_RO | 0x0C00 | 32 | U-Box PMON Global Control |

2.1.6.1 Global PMON Global Control/Status Registers

The following registers represent state governing all PMUs in the uncore, both to exert global control and collect box-level information.

`U_MSR_PMON_GLOBAL_CTL` contains bits that can reset (`.rst_all`) and freeze/enable (`.en_all`) all the uncore counters. The `.en_all` bit must be set to 1 before any uncore counters will collect events.

Note: The register also contains the enable for the U-Box counters.

If an overflow is detected in any of the uncore's PMON registers, it will be summarized in `U_MSR_PMON_GLOBAL_STATUS`. This register accumulates overflows sent to it from the U-Box, W-Box and S-Boxes and indicates if a disable was received from one of the boxes. To reset the summary overflow bits, a user must set the corresponding bits in the `U_MSR_PMON_GLOBAL_OVF_CTL` register.

Table 2-2. U_MSR_PMON_GLOBAL_CTL Register – Field Definitions

| Field | Bits | HW Reset Value | Description |
|--------------|-------|----------------|---|
| frz_all | 31 | 0 | Disable uncore counting (by clearing .en_all) if PMI is received from box with overflowing counter. |
| ig | 30 | 0 | Read zero; writes ignored. |
| rst_all | 29 | 0 | Reset All Uncore PMON Counters |
| en_all | 28 | 0 | Enable All Uncore PMON Counters |
| ig | 27:11 | 0 | Read zero; writes ignored. |
| pmi_core_sel | 10:1 | 0 | PMI Core Select Ex: If counter pmi is sent to U-Box for Box with overflowing counter... 00000000 - No PMI sent 00000001 - Send PMI to core 0 10000000 - Send PMI to core 7 11000100 - Send PMI to core 2, 6 & 7 etc. |
| en | 0 | 0 | Enable U-Box PMON counters |

Table 2-3. U_MSR_PMON_GLOBAL_STATUS Register – Field Definitions

| Field | Bits | HW Reset Value | Description |
|-------|------|----------------|---|
| cond | 31 | 0 | Condition Change. |
| pmi | 30 | 0 | PMI Received from box with overflowing counter. |
| ig | 29:4 | 0 | Read zero; writes ignored. |
| ov_s0 | 3 | 0 | Set if overflow is detected from a S-Box 0 PMON register. |
| ov_s1 | 2 | 0 | Set if overflow is detected from a S-Box 1 PMON register. |
| ov_w | 1 | 0 | Set if overflow is detected from a W-Box PMON register. |
| ov_u | 0 | 0 | Set if overflow is detected from a U-Box PMON register. |

Table 2-4. U_MSR_PMON_GLOBAL_OVF_CTL Register – Field Definitions

| Field | Bits | HW Reset Value | Description |
|-----------|------|----------------|----------------------------|
| clr_cond | 31 | 0 | Clear Condition Change |
| clr_pmi | 30 | 0 | Clear PMI Received bit. |
| ig | 29:4 | 0 | Read zero; writes ignored. |
| clr_ov_s0 | 3 | 0 | Clear S-Box 0 Overflow |
| clr_ov_s1 | 2 | 0 | Clear S-Box 1 Overflow |
| clr_ov_w | 1 | 0 | Clear W-Box Overflow |
| clr_ov_u | 0 | 0 | Clear U-Box Overflow |

2.2 U-Box Performance Monitoring

The U-Box serves as the system configuration controller for the Intel® Xeon® Processor E7 Family. It contains one counter which can be configured to capture a small set of events.

2.2.1 U-Box PMON Summary

Table 2-5. U-Box Performance Monitoring MSRs

| MSR Name | Access | MSR Address | Size (bits) | Description |
|-------------------|--------|-------------|-------------|-------------------------|
| U_MSR_PMON_CTR | RW_RW | 0x0C11 | 64 | U-Box PMON Counter |
| U_MSR_PMON_EV_SEL | RW_RO | 0x0C10 | 32 | U-Box PMON Event Select |

2.2.1.1 U-Box Box Level PMON State

U-Box global state bits are stored in the uncore global state registers. Refer to [Section 2.1, “Global Performance Monitoring Control Counter Overflow”](#) for more information.

2.2.1.2 U-Box PMON state - Counter/Control Pairs

The following table defines the layout of the U-Box performance monitor control register. The main task of this configuration register is to select the event to be monitored by its respective data counter. Setting the `.ev_sel` field performs the event selection. The `.en` bit must be set to 1 to enable counting.

Additional control bits include:

- `.pmi_en` which governs what to do if an overflow is detected.
- `.edge_detect` - Rather than accumulating the raw count each cycle, the register can capture transitions from no event to an event incoming.

Table 2-6. U_MSR_PMON_EVT_SEL Register – Field Definitions

| Field | Bits | HW Reset Value | Description |
|-------------|-------|----------------|--|
| ig | 63 | 0 | Read zero; writes ignored. |
| rsv | 62 | 0 | Reserved; Must write to 0 else behavior is undefined. |
| ig | 61:23 | 0 | Read zero; writes ignored. |
| en | 22 | 0 | Local Counter Enable. When set, the associated counter is locally enabled. NOTE: It must also be enabled in U_MSR_PMON_GLOBAL_CTL and the U-Box to be fully enabled. |
| ig | 21 | 0 | Read zero; writes ignored. |
| pmi_en | 20 | 0 | When this bit is asserted and the corresponding counter overflows, a PMI exception is sent to the U-Box. |
| ig | 19 | 0 | Read zero; writes ignored. |
| edge_detect | 18 | 0 | When asserted, the 0 to 1 transition edge of a 1 bit event input will cause the corresponding counter to increment. When 0, the counter will increment for however long the event is asserted. |
| ig | 17:8 | 0 | Read zero; writes ignored. |
| ev_sel | 7:0 | 0 | Select event to be counted. |

The U-Box performance monitor data register is 48b wide. A counter overflow occurs when a carry out bit from bit 47 is detected. Software can force all uncore counting to freeze after N events by preloading a monitor with a count value of $2^{48} - N$ and setting the control register to send a PMI to the U-Box. Upon receipt of the PMI, the U-Box will disable counting ([Section 2.1.1, "Freezing on Counter Overflow"](#)). During the interval of time between overflow and global disable, the counter value will wrap and continue to collect events.

In this way, software can capture the precise number of events that occurred between the time uncore counting was enabled and when it was disabled (or 'frozen') with minimal skew.

If accessible, software can continuously read the data registers without disabling event collection.

Table 2-7. U_MSR_PMON_CTR Register – Field Definitions

| Field | Bits | HW Reset Value | Description |
|-------------|------|----------------|----------------------------------|
| event_count | 47:0 | 0 | 48-bit performance event counter |

2.2.2 U-Box Performance Monitoring Events

The set of events that can be monitored in the U-Box are summarized in the following section.

- **Tracks NcMsgS packets** generated by the U-Box, as they arbitrate to be broadcast. They are prioritized as follows: Special Cycle->StopReq1/StartReq2->Lock/Unlock->Remote Interrupts->Local Interrupts.
- **Errors** detected and distinguished between recoverable, corrected, uncorrected and fatal.
- Number of times cores were sent **IPIs** or were **Woken** up.

- **Requests** to the Ring or a B-Box.

etc.

2.2.3 U-Box Events Ordered By Code

Table 2-8 summarizes the directly-measured U-Box events.

Table 2-8. Performance Monitor Events for U-Box Events

| Symbol Name | Event Code | Max Inc/Cyc | Description |
|----------------------|------------|-------------|----------------------------------|
| BUF_VALID_LOCAL_INT | 0x000 | 1 | Local IPI Buffer is valid |
| BUF_VALID_REMOTE_INT | 0x001 | 1 | Remote IPI Buffer is valid |
| BUF_VALID_LOCK | 0x002 | 1 | Lock Buffer is valid |
| BUF_VALID_STST | 0x003 | 1 | Start/Stop Req Buffer is valid |
| BUF_VALID_SPC_CYCLES | 0x004 | 1 | SpCyc Buffer is valid |
| BUF_VALID_DOOR_BELL | 0x005 | 1 | Door Bell Buffer Valid |
| BUF_VALID_PM_IN | 0x006 | 1 | PM Request In Buffer Valid |
| BUF_VALID_PM_OUT | 0x007 | 1 | PM Request Out Buffer Valid |
| U2R_REQUESTS | 0x050 | 1 | Number U-Box to Ring Requests |
| U2B_REQUEST_CYCLES | 0x051 | 1 | U to B-Box Active Request Cycles |
| WOKEN | 0x0F8 | 1 | Number of core woken up |
| IPIS_SENT | 0x0F9 | 1 | Number of core IPIs sent |
| RECOV | 0x1DF | 1 | Recoverable |
| CORRECTED_ERR | 0x1E4 | 1 | Corrected Error |
| UNCORRECTED_ERR | 0x1E5 | 1 | Uncorrected Error |
| FATAL_ERR | 0x1E6 | 1 | Fatal Error |

2.2.4 U-Box Performance Monitor Event List

This section enumerates Intel® Xeon® Processor E7 Family uncore performance monitoring events for the U-Box.

BUF_VALID_DOOR_BELL

- **Title:** Door Bell Buffer Valid
- **Category:** U-Box Events
- **Event Code:** 0x005, **Max. Inc/Cyc:** 1,
- **Definition:** Number of uncore cycles the Door Bell packet buffer contained a valid entry.

BUF_VALID_LOCAL_INT

- **Title:** Local IPI Buffer Valid
- **Category:** U-Box Events
- **Event Code:** 0x000, **Max. Inc/Cyc:** 1,
- **Definition:** Number of cycles the Local Interrupt packet buffer contained a valid entry.

BUF_VALID_LOCK

- **Title:** Lock Buffer Valid
- **Category:** U-Box Events
- **Event Code:** 0x002, **Max. Inc/Cyc:** 1,
- **Definition:** Number of cycles the Lock packet buffer contained a valid entry.

BUF_VALID_PMI_IN

- **Title:** PM Request In Buffer Valid
- **Category:** U-Box Events
- **Event Code:** 0x006, **Max. Inc/Cyc:** 1,
- **Definition:** Number of uncore cycles the PM Request In packet buffer contained a valid entry.

BUF_VALID_PMI_OUT

- **Title:** PM Request Out Buffer Valid
- **Category:** U-Box Events
- **Event Code:** 0x007, **Max. Inc/Cyc:** 1,
- **Definition:** Number of uncore cycles the PM Request Out packet buffer contained a valid entry.

BUF_VALID_REMOTE_INT

- **Title:** Remote IPI Buffer Valid
- **Category:** U-Box Events
- **Event Code:** 0x001, **Max. Inc/Cyc:** 1,
- **Definition:** Number of cycles the Remote IPI packet buffer contained a valid entry.

BUF_VALID_SPC_CYCLES

- **Title:** SpcCyc Buffer Valid
- **Category:** U-Box Events
- **Event Code:** 0x004, **Max. Inc/Cyc:** 1,
- **Definition:** Number of uncore cycles the Special Cycle packet buffer contains a valid entry. 'Special Cycles' are NcMsgS packets generated by the U-Box and broadcast to internal cores to cover such things as Shutdown, Invd_Ack and WbInvd_Ack conditions.

BUF_VALID_STST

- **Title:** Start/Stop Req Buffer Valid
- **Category:** U-Box Events
- **Event Code:** 0x003, **Max. Inc/Cyc:** 1,
- **Definition:** Number of uncore cycles the Start/Stop Request packet buffer contained a valid entry.

CORRECTED_ERR

- **Title:** Corrected Errors
- **Category:** U-Box Events
- **Event Code:** 0x1E4, **Max. Inc/Cyc:** 1,
- **Definition:** Number of corrected errors.

FATAL_ERR

- **Title:** Fatal Errors
- **Category:** U-Box Events
- **Event Code:** 0x1E6, **Max. Inc/Cyc:** 1,
- **Definition:** Number of fatal errors.

IPIS_SENT

- **Title:** Number Core IPIs Sent
- **Category:** U-Box Events
- **Event Code:** 0x0F9, **Max. Inc/Cyc:** 1,
- **Definition:** Number of core IPIs sent.

RECOV

- **Title:** Recoverable
- **Category:** U-Box Events
- **Event Code:** 0x1DF, **Max. Inc/Cyc:** 1,
- **Definition:** Number of recoverable errors.

U2R_REQUESTS

- **Title:** Number U2R Requests
- **Category:** U-Box Events
- **Event Code:** 0x050, **Max. Inc/Cyc:** 1,
- **Definition:** Number U-Box to Ring Requests.

U2B_REQUEST_CYCLES

- **Title:** U2B Active Request Cycles
- **Category:** U-Box Events
- **Event Code:** 0x051, **Max. Inc/Cyc:** 1,
- **Definition:** Number U to B-Box Active Request Cycles.

UNCORRECTED_ERR

- **Title:** Uncorrected Error
- **Category:** U-Box Events
- **Event Code:** 0x1E5, **Max. Inc/Cyc:** 1,
- **Definition:** Number of uncorrected errors.

WOKEN

- **Title:** Number Cores Woken Up
- **Category:** U-Box Events
- **Event Code:** 0x0F8, **Max. Inc/Cyc:** 1,
- **Definition:** Number of cores woken up.

2.3 C-Box Performance Monitoring

2.3.1 Overview of the C-Box

For the Intel® Xeon® Processor E7 Family, the LLC coherence engine (C-Box) manages the interface between the core and the last level cache (LLC). All core transactions that access the LLC are directed from the core to a C-Box via the ring interconnect. The C-Box is responsible for managing data delivery from the LLC to the requesting core. It is also responsible for maintaining coherence between the cores within the socket that share the LLC; generating snoops and collecting snoop responses to the local cores when the MESI protocol requires it.

The C-Box is also the gate keeper for all Intel® QuickPath Interconnect (Intel® QPI) messages that originate in the core and is responsible for ensuring that all Intel QuickPath Interconnect messages that pass through the socket's LLC remain coherent.

The Intel® Xeon® Processor E7 Family contains eight instances of the C-Box, each assigned to manage a distinct 3MB, 24-way set associative slice of the processor's total LLC capacity. For processors with fewer than 10 3MB LLC slices, the C-Boxes for missing slices will still be active and track ring traffic caused by their co-located core even if they have no LLC related traffic to track (i.e. hits/misses/snoops).

Every physical memory address in the system is uniquely associated with a single C-Box instance via a proprietary hashing algorithm that is designed to keep the distribution of traffic across the C-Box instances relatively uniform for a wide range of possible address patterns. This enables the individual C-Box instances to operate independently, each managing its slice of the physical address space without any C-Box in a given socket ever needing to communicate with the other C-Boxes in that same socket.

Each C-Box is uniquely associated with a single S-Box. All messages which a given C-Box sends out to the system memory or Intel QPI pass through the S-Box that is physically closest to that C-Box.

2.3.2 C-Box Performance Monitoring Overview

Each of the C-Boxes in the Intel® Xeon® Processor E7 Family supports event monitoring through six 48-bit wide counters (CBx_CR_C_MSR_PMON_CTR{5:0}). Each of these six counters can be programmed to count any C-Box event. The C-Box counters can increment by a maximum of 5b per cycle.

For information on how to setup a monitoring session, refer to [Section 2.1, “Global Performance Monitoring Control Counter Overflow”](#).

2.3.2.1 C-Box PMU - Overflow, Freeze and Unfreeze

If an overflow is detected from a C-Box performance counter, the overflow bit is set at the box level (C_MSR_PMON_GLOBAL_STATUS.ov), and forwarded up the chain towards the U-Box. If a C-Box0 counter overflows, a notification is sent and stored in S-Box0 (S_MSR_PMON_SUMMARY.ov_c_l) which, in turn, sends the overflow notification up to the U-Box (U_MSR_PMON_GLOBAL_STATUS.ov_s0). Refer to [Table 2-26, “S_MSR_PMON_SUMMARY Register Fields”](#) to determine how each C-Box’s overflow bit is accumulated in the attached S-Box.

HW can be also configured (by setting the corresponding *.pmi_en* to 1) to send a PMI to the U-Box when an overflow is detected. The U-Box may be configured to freeze all uncore counting and/or send a PMI to selected cores when it receives this signal.

Once a freeze has occurred, in order to see a new freeze, the overflow field responsible for the freeze, must be cleared by setting the corresponding bit in C_MSR_PMON_GLOBAL_OVF_CTL.clr_ov. Assuming all the counters have been locally enabled (.en bit in data registers meant to monitor events) and the overflow bit(s) has been cleared, the C-Box is prepared for a new sample interval. Once the global controls have been re-enabled ([Section 2.1.5, “Enabling a New Sample Interval from Frozen Counters”](#)), counting will resume.

2.3.3 C-BOX Performance Monitors

Table 2-9. C-Box Performance Monitoring MSRs

| MSR Name | Access | MSR Address | Size (bits) | Description |
|----------------------------------|--------|-------------|-------------|--------------------------------------|
| CB9_CR_C_MSR_PMON_CTR_5 | RW_RW | 0xFDB | 64 | C-Box 9 PMON Counter 5 |
| CB9_CR_C_MSR_PMON_EVT_SEL_5 | RW_RO | 0xFDA | 64 | C-Box 9 PMON Event Select 5 |
| CB9_CR_C_MSR_PMON_CTR_4 | RW_RW | 0xFD9 | 64 | C-Box 9 PMON Counter 4 |
| CB9_CR_C_MSR_PMON_EVT_SEL_4 | RW_RO | 0xFD8 | 64 | C-Box 9 PMON Event Select 4 |
| CB9_CR_C_MSR_PMON_CTR_3 | RW_RW | 0xFD7 | 64 | C-Box 9 PMON Counter 3 |
| CB9_CR_C_MSR_PMON_EVT_SEL_3 | RW_RO | 0xFD6 | 64 | C-Box 9 PMON Event Select 3 |
| CB9_CR_C_MSR_PMON_CTR_2 | RW_RW | 0xFD5 | 64 | C-Box 9 PMON Counter 2 |
| CB9_CR_C_MSR_PMON_EVT_SEL_2 | RW_RO | 0xFD4 | 64 | C-Box 9 PMON Event Select 2 |
| CB9_CR_C_MSR_PMON_CTR_1 | RW_RW | 0xFD3 | 64 | C-Box 9 PMON Counter 1 |
| CB9_CR_C_MSR_PMON_EVT_SEL_1 | RW_RO | 0xFD2 | 64 | C-Box 9 PMON Event Select 1 |
| CB9_CR_C_MSR_PMON_CTR_0 | RW_RW | 0xFD1 | 64 | C-Box 9 PMON Counter 0 |
| CB9_CR_C_MSR_PMON_EVT_SEL_0 | RW_RO | 0xFD0 | 64 | C-Box 9 PMON Event Select 0 |
| CB9_CR_C_MSR_PMON_GLOBAL_OVF_CTL | WO_RO | 0xFC2 | 32 | C-Box 9 PMON Global Overflow Control |
| CB9_CR_C_MSR_PMON_GLOBAL_STATUS | RW_RW | 0xFC1 | 32 | C-Box 9 PMON Global Status |
| CB9_CR_C_MSR_PMON_GLOBAL_CTL | RW_RO | 0xFC0 | 32 | C-Box 9 PMON Global Control |
| | | | | |
| CB8_CR_C_MSR_PMON_CTR_5 | RW_RW | 0xF5B | 64 | C-Box 8 PMON Counter 5 |
| CB8_CR_C_MSR_PMON_EVT_SEL_5 | RW_RO | 0xF5A | 64 | C-Box 8 PMON Event Select 5 |
| CB8_CR_C_MSR_PMON_CTR_4 | RW_RW | 0xF59 | 64 | C-Box 8 PMON Counter 4 |
| CB8_CR_C_MSR_PMON_EVT_SEL_4 | RW_RO | 0xF58 | 64 | C-Box 8 PMON Event Select 4 |
| CB8_CR_C_MSR_PMON_CTR_3 | RW_RW | 0xF57 | 64 | C-Box 8 PMON Counter 3 |
| CB8_CR_C_MSR_PMON_EVT_SEL_3 | RW_RO | 0xF56 | 64 | C-Box 8 PMON Event Select 3 |
| CB8_CR_C_MSR_PMON_CTR_2 | RW_RW | 0xF55 | 64 | C-Box 8 PMON Counter 2 |
| CB8_CR_C_MSR_PMON_EVT_SEL_2 | RW_RO | 0xF54 | 64 | C-Box 8 PMON Event Select 2 |
| CB8_CR_C_MSR_PMON_CTR_1 | RW_RW | 0xF53 | 64 | C-Box 8 PMON Counter 1 |
| CB8_CR_C_MSR_PMON_EVT_SEL_1 | RW_RO | 0xF52 | 64 | C-Box 8 PMON Event Select 1 |
| CB8_CR_C_MSR_PMON_CTR_0 | RW_RW | 0xF51 | 64 | C-Box 8 PMON Counter 0 |
| CB8_CR_C_MSR_PMON_EVT_SEL_0 | RW_RO | 0xF50 | 64 | C-Box 8 PMON Event Select 0 |
| CB8_CR_C_MSR_PMON_GLOBAL_OVF_CTL | WO_RO | 0xF42 | 32 | C-Box 8 PMON Global Overflow Control |
| CB8_CR_C_MSR_PMON_GLOBAL_STATUS | RW_RW | 0xF41 | 32 | C-Box 8 PMON Global Status |
| CB8_CR_C_MSR_PMON_GLOBAL_CTL | RW_RO | 0xF40 | 32 | C-Box 8 PMON Global Control |
| | | | | |
| CB7_CR_C_MSR_PMON_CTR_5 | RW_RW | 0xDFB | 64 | C-Box 7 PMON Counter 5 |
| CB7_CR_C_MSR_PMON_EVT_SEL_5 | RW_RO | 0xDFA | 64 | C-Box 7 PMON Event Select 5 |
| CB7_CR_C_MSR_PMON_CTR_4 | RW_RW | 0xDF9 | 64 | C-Box 7 PMON Counter 4 |
| CB7_CR_C_MSR_PMON_EVT_SEL_4 | RW_RO | 0xDF8 | 64 | C-Box 7 PMON Event Select 4 |
| CB7_CR_C_MSR_PMON_CTR_3 | RW_RW | 0xDF7 | 64 | C-Box 7 PMON Counter 3 |
| CB7_CR_C_MSR_PMON_EVT_SEL_3 | RW_RO | 0xDF6 | 64 | C-Box 7 PMON Event Select 3 |
| CB7_CR_C_MSR_PMON_CTR_2 | RW_RW | 0xDF5 | 64 | C-Box 7 PMON Counter 2 |

| MSR Name | Access | MSR Address | Size (bits) | Description |
|----------------------------------|--------|-------------|-------------|--------------------------------------|
| CB7_CR_C_MSR_PMON_EVT_SEL_2 | RW_RO | 0xDF4 | 64 | C-Box 7 PMON Event Select 2 |
| CB7_CR_C_MSR_PMON_CTR_1 | RW_RW | 0xDF3 | 64 | C-Box 7 PMON Counter 1 |
| CB7_CR_C_MSR_PMON_EVT_SEL_1 | RW_RO | 0xDF2 | 64 | C-Box 7 PMON Event Select 1 |
| CB7_CR_C_MSR_PMON_CTR_0 | RW_RW | 0xDF1 | 64 | C-Box 7 PMON Counter 0 |
| CB7_CR_C_MSR_PMON_EVT_SEL_0 | RW_RO | 0xDF0 | 64 | C-Box 7 PMON Event Select 0 |
| CB7_CR_C_MSR_PMON_GLOBAL_OVF_CTL | WO_RO | 0xDE2 | 32 | C-Box 7 PMON Global Overflow Control |
| CB7_CR_C_MSR_PMON_GLOBAL_STATUS | RW_RW | 0xDE1 | 32 | C-Box 7 PMON Global Status |
| CB7_CR_C_MSR_PMON_GLOBAL_CTL | RW_RO | 0xDE0 | 32 | C-Box 7 PMON Global Control |
| | | | | |
| CB3_CR_C_MSR_PMON_CTR_5 | RW_RW | 0xDDB | 64 | C-Box 3 PMON Counter 5 |
| CB3_CR_C_MSR_PMON_EVT_SEL_5 | RW_RO | 0xDDA | 64 | C-Box 3 PMON Event Select 5 |
| CB3_CR_C_MSR_PMON_CTR_4 | RW_RW | 0xDD9 | 64 | C-Box 3 PMON Counter 4 |
| CB3_CR_C_MSR_PMON_EVT_SEL_4 | RW_RO | 0xDD8 | 64 | C-Box 3 PMON Event Select 4 |
| CB3_CR_C_MSR_PMON_CTR_3 | RW_RW | 0xDD7 | 64 | C-Box 3 PMON Counter 3 |
| CB3_CR_C_MSR_PMON_EVT_SEL_3 | RW_RO | 0xDD6 | 64 | C-Box 3 PMON Event Select 3 |
| CB3_CR_C_MSR_PMON_CTR_2 | RW_RW | 0xDD5 | 64 | C-Box 3 PMON Counter 2 |
| CB3_CR_C_MSR_PMON_EVT_SEL_2 | RW_RO | 0xDD4 | 64 | C-Box 3 PMON Event Select 2 |
| CB3_CR_C_MSR_PMON_CTR_1 | RW_RW | 0xDD3 | 64 | C-Box 3 PMON Counter 1 |
| CB3_CR_C_MSR_PMON_EVT_SEL_1 | RW_RO | 0xDD2 | 64 | C-Box 3 PMON Event Select 1 |
| CB3_CR_C_MSR_PMON_CTR_0 | RW_RW | 0xDD1 | 64 | C-Box 3 PMON Counter 0 |
| CB3_CR_C_MSR_PMON_EVT_SEL_0 | RW_RO | 0xDD0 | 64 | C-Box 3 PMON Event Select 0 |
| CB3_CR_C_MSR_PMON_GLOBAL_OVF_CTL | WO_RO | 0xDC2 | 32 | C-Box 3 PMON Global Overflow Control |
| CB3_CR_C_MSR_PMON_GLOBAL_STATUS | RW_RW | 0xDC1 | 32 | C-Box 3 PMON Global Status |
| CB3_CR_C_MSR_PMON_GLOBAL_CTL | RW_RO | 0xDC0 | 32 | C-Box 3 PMON Global Control |
| | | | | |
| CB5_CR_C_MSR_PMON_CTR_5 | RW_RW | 0xDBB | 64 | C-Box 5 PMON Counter 5 |
| CB5_CR_C_MSR_PMON_EVT_SEL_5 | RW_RO | 0xDBA | 64 | C-Box 5 PMON Event Select 5 |
| CB5_CR_C_MSR_PMON_CTR_4 | RW_RW | 0xDB9 | 64 | C-Box 5 PMON Counter 4 |
| CB5_CR_C_MSR_PMON_EVT_SEL_4 | RW_RO | 0xDB8 | 64 | C-Box 5 PMON Event Select 4 |
| CB5_CR_C_MSR_PMON_CTR_3 | RW_RW | 0xDB7 | 64 | C-Box 5 PMON Counter 3 |
| CB5_CR_C_MSR_PMON_EVT_SEL_3 | RW_RO | 0xDB6 | 64 | C-Box 5 PMON Event Select 3 |
| CB5_CR_C_MSR_PMON_CTR_2 | RW_RW | 0xDB5 | 64 | C-Box 5 PMON Counter 2 |
| CB5_CR_C_MSR_PMON_EVT_SEL_2 | RW_RO | 0xDB4 | 64 | C-Box 5 PMON Event Select 2 |
| CB5_CR_C_MSR_PMON_CTR_1 | RW_RW | 0xDB3 | 64 | C-Box 5 PMON Counter 1 |
| CB5_CR_C_MSR_PMON_EVT_SEL_1 | RW_RO | 0xDB2 | 64 | C-Box 5 PMON Event Select 1 |
| CB5_CR_C_MSR_PMON_CTR_0 | RW_RW | 0xDB1 | 64 | C-Box 5 PMON Counter 0 |
| CB5_CR_C_MSR_PMON_EVT_SEL_0 | RW_RO | 0xDB0 | 64 | C-Box 5 PMON Event Select 0 |
| CB5_CR_C_MSR_PMON_GLOBAL_OVF_CTL | WO_RO | 0xDA2 | 32 | C-Box 5 PMON Global Overflow Control |
| CB5_CR_C_MSR_PMON_GLOBAL_STATUS | RW_RW | 0xDA1 | 32 | C-Box 5 PMON Global Status |
| CB5_CR_C_MSR_PMON_GLOBAL_CTL | RW_RO | 0xDA0 | 32 | C-Box 5 PMON Global Control |
| | | | | |

| MSR Name | Access | MSR Address | Size (bits) | Description |
|----------------------------------|--------|-------------|-------------|--------------------------------------|
| CB1_CR_C_MSR_PMON_CTR_5 | RW_RW | 0xD9B | 64 | C-Box 1 PMON Counter 5 |
| CB1_CR_C_MSR_PMON_EVT_SEL_5 | RW_RO | 0xD9A | 64 | C-Box 1 PMON Event Select 5 |
| CB1_CR_C_MSR_PMON_CTR_4 | RW_RW | 0xD99 | 64 | C-Box 1 PMON Counter 4 |
| CB1_CR_C_MSR_PMON_EVT_SEL_4 | RW_RO | 0xD98 | 64 | C-Box 1 PMON Event Select 4 |
| CB1_CR_C_MSR_PMON_CTR_3 | RW_RW | 0xD97 | 64 | C-Box 1 PMON Counter 3 |
| CB1_CR_C_MSR_PMON_EVT_SEL_3 | RW_RO | 0xD96 | 64 | C-Box 1 PMON Event Select 3 |
| CB1_CR_C_MSR_PMON_CTR_2 | RW_RW | 0xD95 | 64 | C-Box 1 PMON Counter 2 |
| CB1_CR_C_MSR_PMON_EVT_SEL_2 | RW_RO | 0xD94 | 64 | C-Box 1 PMON Event Select 2 |
| CB1_CR_C_MSR_PMON_CTR_1 | RW_RW | 0xD93 | 64 | C-Box 1 PMON Counter 1 |
| CB1_CR_C_MSR_PMON_EVT_SEL_1 | RW_RO | 0xD92 | 64 | C-Box 1 PMON Event Select 1 |
| CB1_CR_C_MSR_PMON_CTR_0 | RW_RW | 0xD91 | 64 | C-Box 1 PMON Counter 0 |
| CB1_CR_C_MSR_PMON_EVT_SEL_0 | RW_RO | 0xD90 | 64 | C-Box 1 PMON Event Select 0 |
| CB1_CR_C_MSR_PMON_GLOBAL_OVF_CTL | WO_RO | 0xD82 | 32 | C-Box 1 PMON Global Overflow Control |
| CB1_CR_C_MSR_PMON_GLOBAL_STATUS | RW_RW | 0xD81 | 32 | C-Box 1 PMON Global Status |
| CB1_CR_C_MSR_PMON_GLOBAL_CTL | RW_RO | 0xD80 | 32 | C-Box 1 PMON Global Control |
| | | | | |
| CB6_CR_C_MSR_PMON_CTR_5 | RW_RW | 0xD7B | 64 | C-Box 6 PMON Counter 5 |
| CB6_CR_C_MSR_PMON_EVT_SEL_5 | RW_RO | 0xD7A | 64 | C-Box 6 PMON Event Select 5 |
| CB6_CR_C_MSR_PMON_CTR_4 | RW_RW | 0xD79 | 64 | C-Box 6 PMON Counter 4 |
| CB6_CR_C_MSR_PMON_EVT_SEL_4 | RW_RO | 0xD78 | 64 | C-Box 6 PMON Event Select 4 |
| CB6_CR_C_MSR_PMON_CTR_3 | RW_RW | 0xD77 | 64 | C-Box 6 PMON Counter 3 |
| CB6_CR_C_MSR_PMON_EVT_SEL_3 | RW_RO | 0xD76 | 64 | C-Box 6 PMON Event Select 3 |
| CB6_CR_C_MSR_PMON_CTR_2 | RW_RW | 0xD75 | 64 | C-Box 6 PMON Counter 2 |
| CB6_CR_C_MSR_PMON_EVT_SEL_2 | RW_RO | 0xD74 | 64 | C-Box 6 PMON Event Select 2 |
| CB6_CR_C_MSR_PMON_CTR_1 | RW_RW | 0xD73 | 64 | C-Box 6 PMON Counter 1 |
| CB6_CR_C_MSR_PMON_EVT_SEL_1 | RW_RO | 0xD72 | 64 | C-Box 6 PMON Event Select 1 |
| CB6_CR_C_MSR_PMON_CTR_0 | RW_RW | 0xD71 | 64 | C-Box 6 PMON Counter 0 |
| CB6_CR_C_MSR_PMON_EVT_SEL_0 | RW_RO | 0xD70 | 64 | C-Box 6 PMON Event Select 0 |
| CB6_CR_C_MSR_PMON_GLOBAL_OVF_CTL | WO_RO | 0xD62 | 32 | C-Box 6 PMON Global Overflow Control |
| CB6_CR_C_MSR_PMON_GLOBAL_STATUS | RW_RW | 0xD61 | 32 | C-Box 6 PMON Global Status |
| CB6_CR_C_MSR_PMON_GLOBAL_CTL | RW_RO | 0xD60 | 32 | C-Box 6 PMON Global Control |
| | | | | |
| CB2_CR_C_MSR_PMON_CTR_5 | RW_RW | 0xD5B | 64 | C-Box 2 PMON Counter 5 |
| CB2_CR_C_MSR_PMON_EVT_SEL_5 | RW_RO | 0xD5A | 64 | C-Box 2 PMON Event Select 5 |
| CB2_CR_C_MSR_PMON_CTR_4 | RW_RW | 0xD59 | 64 | C-Box 2 PMON Counter 4 |
| CB2_CR_C_MSR_PMON_EVT_SEL_4 | RW_RO | 0xD58 | 64 | C-Box 2 PMON Event Select 4 |
| CB2_CR_C_MSR_PMON_CTR_3 | RW_RW | 0xD57 | 64 | C-Box 2 PMON Counter 3 |
| CB2_CR_C_MSR_PMON_EVT_SEL_3 | RW_RO | 0xD56 | 64 | C-Box 2 PMON Event Select 3 |
| CB2_CR_C_MSR_PMON_CTR_2 | RW_RW | 0xD55 | 64 | C-Box 2 PMON Counter 2 |
| CB2_CR_C_MSR_PMON_EVT_SEL_2 | RW_RO | 0xD54 | 64 | C-Box 2 PMON Event Select 2 |
| CB2_CR_C_MSR_PMON_CTR_1 | RW_RW | 0xD53 | 64 | C-Box 2 PMON Counter 1 |

| MSR Name | Access | MSR Address | Size (bits) | Description |
|----------------------------------|--------|-------------|-------------|--------------------------------------|
| CB2_CR_C_MSR_PMON_EVT_SEL_1 | RW_RO | 0xD52 | 64 | C-Box 2 PMON Event Select 1 |
| CB2_CR_C_MSR_PMON_CTR_0 | RW_RW | 0xD51 | 64 | C-Box 2 PMON Counter 0 |
| CB2_CR_C_MSR_PMON_EVT_SEL_0 | RW_RO | 0xD50 | 64 | C-Box 2 PMON Event Select 0 |
| CB2_CR_C_MSR_PMON_GLOBAL_OVF_CTL | WO_RO | 0xD42 | 32 | C-Box 2 PMON Global Overflow Control |
| CB2_CR_C_MSR_PMON_GLOBAL_STATUS | RW_RW | 0xD41 | 32 | C-Box 2 PMON Global Status |
| CB2_CR_C_MSR_PMON_GLOBAL_CTL | RW_RO | 0xD40 | 32 | C-Box 2 PMON Global Control |
| | | | | |
| CB4_CR_C_MSR_PMON_CTR_5 | RW_RW | 0xD3B | 64 | C-Box 4 PMON Counter 5 |
| CB4_CR_C_MSR_PMON_EVT_SEL_5 | RW_RO | 0xD3A | 64 | C-Box 4 PMON Event Select 5 |
| CB4_CR_C_MSR_PMON_CTR_4 | RW_RW | 0xD39 | 64 | C-Box 4 PMON Counter 4 |
| CB4_CR_C_MSR_PMON_EVT_SEL_4 | RW_RO | 0xD38 | 64 | C-Box 4 PMON Event Select 4 |
| CB4_CR_C_MSR_PMON_CTR_3 | RW_RW | 0xD37 | 64 | C-Box 4 PMON Counter 3 |
| CB4_CR_C_MSR_PMON_EVT_SEL_3 | RW_RO | 0xD36 | 64 | C-Box 4 PMON Event Select 3 |
| CB4_CR_C_MSR_PMON_CTR_2 | RW_RW | 0xD35 | 64 | C-Box 4 PMON Counter 2 |
| CB4_CR_C_MSR_PMON_EVT_SEL_2 | RW_RO | 0xD34 | 64 | C-Box 4 PMON Event Select 2 |
| CB4_CR_C_MSR_PMON_CTR_1 | RW_RW | 0xD33 | 64 | C-Box 4 PMON Counter 1 |
| CB4_CR_C_MSR_PMON_EVT_SEL_1 | RW_RO | 0xD32 | 64 | C-Box 4 PMON Event Select 1 |
| CB4_CR_C_MSR_PMON_CTR_0 | RW_RW | 0xD31 | 64 | C-Box 4 PMON Counter 0 |
| CB4_CR_C_MSR_PMON_EVT_SEL_0 | RW_RO | 0xD30 | 64 | C-Box 4 PMON Event Select 0 |
| CB4_CR_C_MSR_PMON_GLOBAL_OVF_CTL | WO_RO | 0xD22 | 32 | C-Box 4 PMON Global Overflow Control |
| CB4_CR_C_MSR_PMON_GLOBAL_STATUS | RW_RW | 0xD21 | 32 | C-Box 4 PMON Global Status |
| CB4_CR_C_MSR_PMON_GLOBAL_CTL | RW_RO | 0xD20 | 32 | C-Box 4 PMON Global Control |
| | | | | |
| CB0_CR_C_MSR_PMON_CTR_5 | RW_RW | 0xD1B | 64 | C-Box 0 PMON Counter 5 |
| CB0_CR_C_MSR_PMON_EVT_SEL_5 | RW_RO | 0xD1A | 64 | C-Box 0 PMON Event Select 5 |
| CB0_CR_C_MSR_PMON_CTR_4 | RW_RW | 0xD19 | 64 | C-Box 0 PMON Counter 4 |
| CB0_CR_C_MSR_PMON_EVT_SEL_4s | RW_RO | 0xD18 | 64 | C-Box 0 PMON Event Select 4 |
| CB0_CR_C_MSR_PMON_CTR_3 | RW_RW | 0xD17 | 64 | C-Box 0 PMON Counter 3 |
| CB0_CR_C_MSR_PMON_EVT_SEL_3 | RW_RO | 0xD16 | 64 | C-Box 0 PMON Event Select 3 |
| CB0_CR_C_MSR_PMON_CTR_2 | RW_RW | 0xD15 | 64 | C-Box 0 PMON Counter 2 |
| CB0_CR_C_MSR_PMON_EVT_SEL_2 | RW_RO | 0xD14 | 64 | C-Box 0 PMON Event Select 2 |
| CB0_CR_C_MSR_PMON_CTR_1 | RW_RW | 0xD13 | 64 | C-Box 0 PMON Counter 1 |
| CB0_CR_C_MSR_PMON_EVT_SEL_1 | RW_RO | 0xD12 | 64 | C-Box 0 PMON Event Select 1 |
| CB0_CR_C_MSR_PMON_CTR_0 | RW_RW | 0xD11 | 64 | C-Box 0 PMON Counter 0 |
| CB0_CR_C_MSR_PMON_EVT_SEL_0 | RW_RO | 0xD10 | 64 | C-Box 0 PMON Event Select 0 |
| CB0_CR_C_MSR_PMON_GLOBAL_OVF_CTL | WO_RO | 0xD02 | 32 | C-Box 0 PMON Global Overflow Control |
| CB0_CR_C_MSR_PMON_GLOBAL_STATUS | RW_RW | 0xD01 | 32 | C-Box 0 PMON Global Status |
| CB0_CR_C_MSR_PMON_GLOBAL_CTL | RW_RO | 0xD00 | 32 | C-Box 0 PMON Global Control |

2.3.3.1 C-Box Box Level PMON state

The following registers represent the state governing all box-level PMUs in the C-Box.

The `_GLOBAL_CTL` register contains the bits used to enable monitoring. It is necessary to set the `.ctr_en` bit to 1 before the corresponding data register can collect events.

If an overflow is detected from one of the C-Box PMON registers, the corresponding bit in the `_GLOBAL_STATUS.ov` field will be set. To reset the overflow bits set in the `_GLOBAL_STATUS.ov` field, a user must set the corresponding bits in the `_GLOBAL_OVF_CTL.clr_ov` field before beginning a new sample interval.

Table 2-10. C_MSR_PMON_GLOBAL_CTL Register – Field Definitions

| Field | Bits | HW Reset Value | Description |
|---------------------|------|----------------|--|
| <code>ctr_en</code> | 5:0 | 0 | Must be set to enable each C-Box counter. NOTE: U-Box enable and per counter enable must also be set to fully enable the counter. |

Table 2-11. C_MSR_PMON_GLOBAL_STATUS Register – Field Definitions

| Field | Bits | HW Reset Value | Description |
|-----------------|------|----------------|---|
| <code>ov</code> | 5:0 | 0 | If an overflow is detected from the corresponding CBOX PMON register, it's overflow bit will be set. NOTE: This bit is also cleared by setting the corresponding bit in <code>C_MSR_PMON_GLOBAL_OVF_CTL</code> |

Table 2-12. C_MSR_PMON_GLOBAL_OVF_CTL Register – Field Definitions

| Field | Bits | HW Reset Value | Description |
|---------------------|------|----------------|--|
| <code>clr_ov</code> | 5:0 | 0 | Write '1' to reset the corresponding <code>C_MSR_PMON_GLOBAL_STATUS</code> overflow bit. |

2.3.3.2 C-Box PMON state - Counter/Control Pairs

The following table defines the layout of the C-Box performance monitor control registers. The main task of these configuration registers is to select the event to be monitored by their respective data counter. Setting the `.ev_sel` and `.umask` fields performs the event selection. The `.en` bit must be set to 1 to enable counting.

Additional control bits include:

- `.pmi_en` governs what to do if an overflow is detected.
- `.threshold` - since C-Box counters can increment by a value greater than 1, a threshold can be applied. If the `.threshold` is set to a non-zero value, that value is compared against the incoming count for that event in each cycle. If the incoming count is \geq the threshold value, then the event count captured in the data register will be incremented by 1.

- *.invert* - Changes the *.threshold* test condition to '<'.
- *.edge_detect* - Rather than accumulating the raw count each cycle (for events that can increment by 1 per cycle), the register can capture transitions from no event to an event incoming.

Table 2-13. C_MSR_PMON_EVT_SEL{5-0} Register – Field Definitions

| Field | Bits | HW Reset Value | Description |
|-------------|-------|----------------|---|
| ig | 63 | 0 | Read zero; writes ignored. |
| rsv | 62:61 | 0 | Reserved; Must write to 0 else behavior is undefined. |
| ig | 60:32 | 0 | Read zero; writes ignored. |
| threshold | 31:24 | 0 | Threshold used in counter comparison. |
| invert | 23 | 0 | When 0, the comparison that will be done is $\text{threshold} \leq \text{event}$. When set to 1, the comparison that is inverted (e.g. $\text{threshold} < \text{event}$) |
| en | 22 | 0 | Local Counter Enable. When set, the associated counter is locally enabled. NOTE: It must also be enabled in C_MSR_PMON_GLOBAL_CTL and the U-Box to be fully enabled. |
| ig | 21 | 0 | Read zero; writes ignored. |
| pmi_en | 20 | 0 | When this bit is asserted and the corresponding counter overflows, a PMI exception is sent to the U-Box. |
| ig | 19 | 0 | Read zero; writes ignored. |
| edge_detect | 18 | 0 | When asserted, the 0 to 1 transition edge of a 1 bit event input will cause the corresponding counter to increment. When 0, the counter will increment for however long the event is asserted. NOTE: <i>.edge_detect</i> is in series following <i>threshold</i> and <i>invert</i> , so it can be applied to multi-increment events that have been filtered by the <i>threshold</i> field. |
| ig | 17:16 | 0 | Read zero; writes ignored. |
| umask | 15:8 | 0 | Select subevents to be counted within the selected event. |
| ev_sel | 7:0 | 0 | Select event to be counted. |

The C-Box performance monitor data registers are 48b wide. A counter overflow occurs when a carry out bit from bit 47 is detected. Software can force all uncore counting to freeze after N events by preloading a monitor with a count value of $2^{48} - N$ and setting the control register to send a PMI to the U-Box. Upon receipt of the PMI, the U-Box will disable counting ([Section 2.1.1, "Freezing on Counter Overflow"](#)). During the interval of time between overflow and global disable, the counter value will wrap and continue to collect events.

In this way, software can capture the precise number of events that occurred between the time uncore counting was enabled and when it was disabled (or 'frozen') with minimal skew.

If accessible, software can continuously read the data registers without disabling event collection.

Table 2-14. C_MSR_PMON_CTR{5-0} Register – Field Definitions

| Field | Bits | HW Reset Value | Description |
|-------------|------|----------------|----------------------------------|
| event_count | 47:0 | 0 | 48-bit performance event counter |

2.3.4 C-BOX Performance Monitoring Events

2.3.4.1 An Overview

The performance monitoring events within the C-Box include all events internal to the LLC as well as events which track ring related activity at the C-Box/Core ring stops. The only ring specific events that are not tracked by the C-Box PMUs are those events that track ring activity at the S-Box ring stop (see the S-Box chapter for details on those events).

C-Box performance monitoring events can be used to track LLC access rates, LLC hit/miss rates, LLC eviction and fill rates, and to detect evidence of back pressure on the LLC pipelines. In addition, the C-Box has performance monitoring events for tracking MESI state transitions that occur as a result of data sharing across sockets in a multi-socket system. And finally, there are events in the C-Box for tracking ring traffic at the C-Box/Core sink inject points.

Every event in the C-Box (with the exception of the P2C inject and *2P sink counts) are from the point of view of the LLC and cannot be associated with any specific core since all cores in the socket send their LLC transactions to all C-Boxes in the socket. The P2C inject and *2P sink counts serve as the exception since those events are tracking ring activity at the cores' ring inject/sink points.

There are separate sets of counters for each C-Box instance. For any event, to get an aggregate count of that event for the entire LLC, the counts across the C-Box instances must be added together. The counts can be averaged across the C-Box instances to get a view of the typical count of an event from the perspective of the individual C-Boxes. Individual per-C-Box deviations from the average can be used to identify hot-spotting across the C-Boxes or other evidences of non-uniformity in LLC behavior across the C-Boxes. Such hot-spotting should be rare, though a repetitive polling on a fixed physical address is one obvious example of a case where an analysis of the deviations across the C-Boxes would indicate hot-spotting.

2.3.4.2 Acronyms frequently used in C-Box Events

The Rings:

AD (Address) Ring - Core Read/Write Requests and Intel QPI Snoops. Carries Intel QPI requests and snoop responses from C to S-Box.

BL (Block or Data) Ring - Data == 2 transfers for 1 cache line

AK (Acknowledge) Ring - Acknowledges S-Box to C-Box and C-Box to Core. Carries snoop responses from Core to C-Box.

IV (Invalidate) Ring - C-Box Snoop requests of core caches

Internal C-Box Queues:

IRQ - Ingress Request Queue on AD Ring. Associated with requests from core.

IPQ - Ingress Probe Queue on AD Ring. Associated with snoops from S-Box.

VIQ - Victim Queue internal to C-Box.

IDQ - Ingress Data Queue on BL Ring. For data from either Core or S-Box.

ICQ - S-Box Ingress Complete Queue on AK Ring

SRQ - Processor Snoop Response Queue on AK ring

IGQ - Ingress GO-pending (tracking GO's to core) Queue

MAF - Miss Address File. Intel QPI ordering buffer that also tracks local coherence.

2.3.4.3 The Queues

There are seven internal occupancy queue counters, each of which is 5bits wide and dedicated to its queue: IRQ, IPQ, VIQ, MAF, RWRF, RSPF, IDF.

Note: IDQ, ICQ, SRQ and IGQ occupancies are not tracked since they are mapped 1:1 to the MAF and, therefore, can not create back pressure.

It should be noted that, while the IRQ, IPQ, VIQ and MAF queues reside within the C-Box; the RWRF, RSPF and IDF queues do not. Instead, they live in-between the Core and the Ring buffering messages as those messages transit between the two. This distinction is useful in that, the queues located within the C-Box can provide information about what is going on in the LLC with respect to the flow of transactions at the point where they become "observed" by the coherence fabric (i.e., where the MAF is located). Occupancy of these buffers informs how many transactions the C-Box is tracking, and where the bottlenecks are manifesting when the C-Box starts to get busy and/or congested.

There is no need to explicitly reset the occupancy counters in the C-Box since they are counting from reset de-assertion.

2.3.4.4 Detecting Performance Problems in the C-Box Pipeline

IRQ occupancy counters should be used to track if the C-Box pipeline is exerting back pressure on the Core-request path. There is a one-to-one correspondence between the LLC requests generated by the cores and the IRQ allocations. IPQ occupancy counters should be used to track if the C-Box pipeline is exerting back pressure on the Intel QPI-snoop path. There is a one-to-one correspondence between the Intel QPI snoops received by the socket, and the IPQ allocations in the C-Boxes. In both cases, if the message is in the IRQ/IPQ then the C-Box hasn't acknowledged it yet and the request hasn't yet entered the LLC's "coherence domain". It deallocates from the IRQ/IPQ at the moment that the C-Box does acknowledge it. In optimal performance scenarios, where there are minimal conflicts between transactions and loads are low enough to keep latencies relatively near to idle, IRQ and IPQ occupancies should remain very low.

One relatively common scenario in which IRQ back pressure will be high is worth mentioning: The IRQ will backup when software is demanding data from memory at a rate that exceeds the available memory BW. The IRQ is designed to be the place where the extra transactions wait U-Box's RTIDs to become available when memory becomes saturated. IRQ back pressure becomes interesting in a scenario where memory is not operating at or near peak sustainable BW. That can be a sign of a performance problem that may be correctable with software tuning.

One final warning on LLC pipeline congestion: Care should be taken not to blindly sum events across C-Boxes without also checking the deviation across individual C-Boxes when investigating performance issues that are concentrated in the C-Box pipelines. Performance problems where congestion in the C-Box pipelines is the cause should be rare, but if they do occur, the event counts may not be homogeneous across the C-Boxes in the socket. The average count across the C-Boxes may be misleading. If performance issues are found in this area it will be useful to know if they are or are not localized to specific C-Boxes.

2.3.5 C-Box Events Ordered By Code

Table 2-15 summarizes the directly-measured C-Box events.

Table 2-15. Performance Monitor Events for C-Box Events

| Symbol Name | Event Code | Max Inc/ Cyc | Description |
|-------------------------------|------------|--------------|---------------------------------------|
| Ring Events | | | |
| BOUNCES_P2C_AD | 0x01 | 1 | Number of P2C AD bounces. |
| BOUNCES_C2P_AK | 0x02 | 1 | Number of C2P AK bounces. |
| BOUNCES_C2P_BL | 0x03 | 1 | Number of C2P BL bounces. |
| BOUNCES_C2P_IV | 0x04 | 1 | Number of C2P IV bounces. |
| SINKS_P2C | 0x05 | 3 | Number of P2C sinks. |
| SINKS_C2P | 0x06 | 3 | Number of C2P sinks. |
| SINKS_S2C | 0x07 | 3 | Number of S2C sinks. |
| SINKS_S2P_BL | 0x08 | 1 | Number of S2P sinks (BL only). |
| ARB_WINS | 0x09 | 7 | Number of ARB wins. |
| ARB_LOSSES | 0x0A | 7 | Number of ARB losses. |
| Local Events | | | |
| STARVED_EGRESS | 0x0B | 8 | Increment on EGR queue starvation |
| EGRESS_BYPASS_WINS | 0x0C | 7 | Egress Bypass Wins |
| INGRESS_BYPASS_WINS_AD | 0x0E | 1 | Ingress S-Box/Non-S-Box Bypass Wins |
| MAF_ACK | 0x10 | 1 | MAF Acknowledges |
| MAF_NACK1 | 0x11 | 1 | MAF Non-Acknowledgements (First Set) |
| MAF_NACK2 | 0x12 | 1 | MAF Non-Acknowledgements (Second Set) |
| LLC_MISSES | 0x14 | 1 | LLC Misses |
| LLC_HITS | 0x15 | 1 | LLC Hits |
| LLC_S_FILLS | 0x16 | 1 | LLC lines filled from S-Box |
| LLC_VICTIMS | 0x17 | 1 | LLC lines victimized |
| Queue Occupancy Events | | | |
| OCCUPANCY_IRQ | 0x18 | 24 | IRQ Occupancy |
| TRANS_IRQ | 0x19 | 1 | IRQ Transactions (dealloc) |
| OCCUPANCY_IPQ | 0x1A | 8 | IPQ Occupancy |
| TRANS_IPQ | 0x1B | 1 | IPQ Transactions |
| OCCUPANCY_VIQ | 0x1C | 8 | VIQ Occupancy |
| TRANS_VIQ | 0x1D | 1 | VIQ Transactions |
| OCCUPANCY_MAF | 0x1E | 16 | MAF Occupancy |
| TRANS_MAF | 0x1F | 1 | MAF Transactions |
| OCCUPANCY_RWRF | 0x20 | 12 | RWRF Occupancy |
| TRANS_RWRF | 0x21 | 1 | RWRF Transactions |
| OCCUPANCY_RSPF | 0x22 | 8 | RSPF Occupancy |
| TRANS_RSPF | 0x23 | 1 | RSPF Transactions |
| SNPS | 0x27 | 1 | Snoops to LLC |
| SNP_HITS | 0x28 | 1 | Snoop Hits in LLC |

Table 2-15. Performance Monitor Events for C-Box Events

| Symbol Name | Event Code | Max Inc/ Cyc | Description |
|----------------------------|------------|-----------------|---|
| Ring Events | | | |
| EGL Special Request | | | |
| SPL_DEAD | 0x29 | 1 | Special Deadlock |
| SPL_ARB_PRI_SW | 0x2A | 1 | Special ARB Priority Switch |
| SPL_NOT_CO | 0x2B | 1 | Special Request Not Collocated |
| SPL_CO_SB | 0x2C | 1 | Special Request Collocated in SB direction |
| SPL_CO_NSB | 0x2D | 1 | Special Request Collocated in NSB direction |
| SPL_IN_FULL_IRQ | 0x2E | 1 | Special Request Allocated in Full IRQ |
| SPL_EGR_SB | 0x2F | 1 | Special Request at EGR in SB direction |
| SPL_EGR_NSB | 0x30 | 1 | Special Request at EGR in NSB direction |
| EGL Miscellaneous | | | |
| IGR_OP_SRAM | 0x31 | 1 | SRAM at IGR Read Output |
| IGR_OP_UC | 0x32 | 1 | UC read at IGR Read Output |
| ARB_WINS_P2C_SB | 0x33 | 1 | P2C ARB Win at EGR in SB direction |
| ARB_WINS_P2C_NSB | 0x34 | 1 | P2C ARB Win at EGR in NSB direction |
| SPOOF_CRD_EMPTY | 0x35 | 1 | Spoof Credit Counter Empty |
| IDF_NONZERO_NO_BL_CRD | 0x36 | 1 | IDF Non-zero but no BL credits |
| IDF_NONZERO_NO_VLD | 0x37 | 1 | IDF Non-zero but no data valid |
| BOUNCE_ASSERT | 0x38 | 1 | IV Bounce Assertion Edge |
| BOUNCE_DEASSERT | 0x39 | 1 | IV Bounce De-assertion Edge |
| SPOOF_ASSERT | 0x3A | 1 | IV Spoof Assertion Edge |
| SPOOF_DEASSERT | 0x3B | 1 | IV Spoof De-assertion Edge |
| IGR_BID_BLOCKED | 0x3C | 1 | IGR Bidding Blocked |

2.3.6 C-Box Performance Monitor Event List

This section enumerates Intel® Xeon® Processor E7 Family uncore performance monitoring events for the C-Box.

ARB_LOSSES

- **Title:** Arbiter Losses.
- **Category:** Ring - Egress
- **Event Code:** 0x0A, **Max. Inc/Cyc:** 7,
- **Definition:** Number of Ring arbitration losses. A loss occurs when a message injection on to the ring fails. This could occur either because there was another message resident on the ring at that ring stop or because the co-located ring agent issued a message onto the ring in the same cycle.

| Extension | umask [15:8] | Description |
|-----------|-----------------|---|
| --- | b00000000 | (*nothing will be counted*) |
| AD_SB | b00000001 | AD ring in the direction that points toward the nearest S-Box |

| Extension | umask [15:8] | Description |
|-----------|-----------------|--|
| AD_NSB | b00000010 | AD ring in the direction that points away from the nearest S-Box |
| AD_ALL | b00000011 | AD ring in either direction. |
| AK_SB | b00000100 | AK ring in the direction that points toward the nearest S-Box |
| AK_NSB | b00001000 | AK ring in the direction that points away from the nearest S-Box |
| AK_ALL | b00001100 | AK ring in either direction. |
| BL_SB | b00010000 | BL ring in the direction that points toward the nearest S-Box |
| BL_NSB | b00100000 | BL ring in the direction that points away from the nearest S-Box |
| BL_ALL | b00110000 | BL ring in either direction. |
| IV | b01000000 | IV ring |
| ALL | b01111111 | All rings |

ARB_WINS

- **Title:** Arbiter Wins
- **Category:** Ring - Egress
- **Event Code:** 0x09, **Max. Inc/Cyc:** 7,
- **Definition:** Number of Ring arbitration wins. A win is when a message was successfully injected onto the ring.

| Extension | umask [15:8] | Description |
|-----------|-----------------|--|
| --- | b00000000 | (*nothing will be counted*) |
| AD_SB | b00000001 | AD ring in the direction that points toward the nearest S-Box |
| AD_NSB | b00000010 | AD ring in the direction that points away from the nearest S-Box |
| AD_ALL | b00000011 | AD ring in either direction. |
| AK_SB | b00000100 | AK ring in the direction that points toward the nearest S-Box |
| AK_NSB | b00001000 | AK ring in the direction that points away from the nearest S-Box |
| AK_ALL | b00001100 | AK ring in either direction. |
| BL_SB | b00010000 | BL ring in the direction that points toward the nearest S-Box |
| BL_NSB | b00100000 | BL ring in the direction that points away from the nearest S-Box |
| BL_ALL | b00110000 | BL ring in either direction. |
| IV | b01000000 | IV ring |
| ALL | b01111111 | All rings |

ARB_WINS_P2C_NSB

- **Title:** P2C ARB Win at EGR in SB direction
- **Category:** EGL Miscellaneous
- **Event Code:** 0x34, **Max. Inc/Cyc:** 1,
- **Definition:** P2C Arb win at C-Box Egress when slot is not reserved by someone else at NSB direction

ARB_WINS_P2C_SB

- **Title:** P2C ARB Win at EGR in SB direction
- **Category:** EGL Miscellaneous
- **Event Code:** 0x33, **Max. Inc/Cyc:** 1,
- **Definition:** P2C Arb win at C-Box Egress when slot is not reserved by someone else at SB direction

BOUNCE_ASSERT

- **Title:** IV Bounce Assertion Edge
- **Category:** EGL Miscellaneous
- **Event Code:** 0x38, **Max. Inc/Cyc:** 1,
- **Definition:** IV bounce assertion edge

BOUNCE_DEASSERT

- **Title:** IV Bounce De-assertion Edge
- **Category:** EGL Miscellaneous
- **Event Code:** 0x39, **Max. Inc/Cyc:** 1,
- **Definition:** IV bounce de-assertion edge

BOUNCES_C2P_AK

- **Title:** C2P AK Bounces
- **Category:** Ring - WIR
- **Event Code:** 0x02, **Max. Inc/Cyc:** 1,
- **Definition:** Number of LLC Ack responses to the core that bounced on the AK ring.

| Extension | umask [15:8] | Description |
|-----------|--------------|---|
| --- | b00000000 | (*nothing will be counted*) |
| SB | b000000x1 | Direction that points toward the nearest S-Box |
| NSB | b0000001x | Direction that points away from the nearest S-Box |
| ALL | b00000011 | Either direction |

BOUNCES_C2P_BL

- **Title:** C2P BL Bounces
- **Category:** Ring - WIR
- **Event Code:** 0x03, **Max. Inc/Cyc:** 1,
- **Definition:** Number of LLC data responses to the core that bounced on the BL ring.

| Extension | umask [15:8] | Description |
|-----------|--------------|---|
| --- | b00000000 | (*nothing will be counted*) |
| SB | b000000x1 | Direction that points toward the nearest S-Box |
| NSB | b0000001x | Direction that points away from the nearest S-Box |
| ALL | b00000011 | Either direction |

BOUNCES_C2P_IV

- **Title:** C2P IV Bounces
- **Category:** Ring - WIR
- **Event Code:** 0x04, **Max. Inc/Cyc:** 1,
- **Definition:** Number of C-Box snoops of a processor's cache that bounced on the IV ring.

BOUNCES_P2C_AD

- **Title:** P2C AD Bounces
- **Category:** Ring - WIR
- **Event Code:** 0x01, **Max. Inc/Cyc:** ,
- **Definition:** Core request to LLC bounces on AD ring.

| Extension | umask [15:8] | Description |
|-----------|--------------|---|
| --- | b00000000 | (*nothing will be counted*) |
| SB | b000000x1 | Direction that points toward the nearest S-Box |
| NSB | b0000001x | Direction that points away from the nearest S-Box |
| ALL | b00000011 | Either direction |

EGRESS_BYPASS_WINS

- **Title:** Egress Bypass Wins
- **Category:** Local - Egress
- **Event Code:** 0x0C, **Max. Inc/Cyc:** 7,
- **Definition:** Number of times a ring egress bypass was taken when a message was injected onto the ring. The subevent field allows tracking of each available egress queue bypass path, including both 0 and 1 cycle versions.

| Extension | umask [15:8] | Description |
|-----------|--------------|-----------------------------|
| --- | b00000000 | (*nothing will be counted*) |
| AD_BYP0 | b00000001 | 0 cycle AD egress bypass |
| AD_BYP1 | b00000010 | 1 cycle AD egress bypass |
| AK_BYP0 | b00000100 | 0 cycle AK egress bypass |
| AK_BYP1 | b00001000 | 1 cycle AK egress bypass |
| BL_BYP0 | b00010000 | 0 cycle BL egress bypass |
| BL_BYP1 | b00100000 | 1 cycle BL egress bypass |
| IV_BYP0 | b01000000 | 0 cycle IV egress bypass |
| IV_BYP1 | b10000000 | 1 cycle IV egress bypass |

IDF_NONZERO_NO_BL_CRD

- **Title:** IDF Non-zero but no BL credits
- **Category:** EGL Miscellaneous
- **Event Code:** 0x36, **Max. Inc/Cyc:** 1,
- **Definition:** IDF count non-zero but no BL credits

IDF_NONZERO_NO_VLD

- **Title:** IDF Non-zero but no data valid
- **Category:** EGL Miscellaneous
- **Event Code:** 0x37, **Max. Inc/Cyc:** 1,
- **Definition:** IDF count non-zero but no data valid (IDF dst is not valid)

IGR_BID_BLOCKED

- **Title:** IGR Bidding Blocked
- **Category:** EGL Miscellaneous
- **Event Code:** 0x3C, **Max. Inc/Cyc:** 1,
- **Definition:** IGR bidding blocked

IGR_OP_SRAM

- **Title:** SRAM at IGR Read Output
- **Category:** EGL Miscellaneous
- **Event Code:** 0x31, **Max. Inc/Cyc:** 1,
- **Definition:** SRAM read at IGR array read output.

IGR_OP_UC

- **Title:** UC read at IGR Read Output
- **Category:** EGL Miscellaneous
- **Event Code:** 0x32, **Max. Inc/Cyc:** 1,
- **Definition:** Uncore read at IGR array read output.

INGRESS_BYPASS_WINS_AD

- **Title:** Ingress S-Box/Non S-Box Bypass Wins
- **Category:** Local - Egress
- **Event Code:** 0x0E, **Max. Inc/Cyc:** 1,
- **Definition:** Number of times that a message, off the AD ring, sunk by the C-Box took one of the ingress queue bypasses. The subevent field allows tracking of each available ingress queue bypass path, including both 0 and 1 cycle versions.

| Extension | umask [15:8] | Description |
|-----------|--------------|--------------------------------------|
| --- | b00000000 | (*nothing will be counted*) |
| IRQ_BYPO | b00000001 | 0 cycle Ingress Request Queue bypass |
| IRQ_BYP1 | b00000010 | 1 cycle Ingress Request Queue bypass |
| IPQ_BYPO | b00000100 | 0 cycle Ingress Probe Queue bypass |
| IPQ_BYP1 | b00001000 | 1 cycle Ingress Probe Queue bypass |

LLC_HITS

- **Title:** LLC Hits
- **Category:** Local - LLC
- **Event Code:** 0x15, **Max. Inc/Cyc:** 1,
- **Definition:** Last Level Cache Hits
- **NOTE:** LRU hints are included in count.

| Extension | umask [15:8] | Description |
|-----------|--------------|--|
| --- | b00000000 | (*nothing will be counted*) |
| M | b0000xxx1 | Modified |
| E | b0000xx1x | Exclusive |
| S | b0000x1xx | Shared |
| F | b00001xxx | Forward (S with right to Forward on snoop) |
| ALL | b00001111 | All hits (to any cacheline state) |

LLC_MISSES

- **Title:** LLC Misses
- **Category:** Local - LLC
- **Event Code:** 0x14, **Max. Inc/Cyc:** 1,
- **Definition:** Last Level Cache Misses

| Extension | umask [15:8] | Description |
|-----------|--------------|---|
| --- | b00000000 | (*nothing will be counted*) |
| S | b00000xx1 | Shared - request requires S line to be upgraded (due to RFO) |
| F | b00000x1x | Forward - request requires F line to be upgraded (due to RFO) |
| I | b000001xx | Invalid - address not found |
| ALL | b00000111 | All misses |

LLC_S_FILLS

- **Title:** LLC S-Box Fills
- **Category:** Local - LLC
- **Event Code:** 0x16, **Max. Inc/Cyc:** 1,
- **Definition:** Last Level Cache lines filled from S-Box

| Extension | umask [15:8] | Description |
|-----------|--------------|--|
| --- | b00000000 | (*nothing will be counted*) |
| M | b0000xxx1 | Filled to LLC in Modified (remote socket forwarded M data without writing back to memory controller) |
| E | b0000xx1x | Filled to LLC in Exclusive |
| S | b0000x1xx | Filled to LLC in Shared |
| F | b00001xxx | Filled to LLC in Forward |
| ALL | b00001111 | All fills to LLC |

LLC_VICTIMS

- **Title:** LLC Lines Victimized
- **Category:** Local - LLC
- **Event Code:** 0x17, **Max. Inc/Cyc:** 1,
- **Definition:** Last Level Cache lines victimized

| Extension | umask [15:8] | Description |
|-----------|--------------|---|
| --- | b00000000 | (*nothing will be counted*) |
| M | b000xxxx1 | Modified data victimized (explicit WB to memory) |
| E | b000xxx1x | Exclusive data victimized |
| S | b000xx1xx | Shared data victimized |
| F | b000x1xxx | Forward data victimized |
| I | b0001xxxx | LLC fill that occurred without victimizing any data |

MAF_ACK

- **Title:** MAF ACK
- **Category:** Local - MAF
- **Event Code:** 0x10, **Max. Inc/Cyc:** 1,
- **Definition:** Miss Address File Acknowledgements.

MAF_NACK1

- **Title:** MAF NACK1
- **Category:** Local - MAF
- **Event Code:** 0x11, **Max. Inc/Cyc:** 1,
- **Definition:** Rejected (not-acknowledged) LLC pipeline passes (Set 1).

| Extension | umask [15:8] | Description |
|-----------------|-----------------|--|
| --- | b00000000 | (*nothing will be counted*) |
| GO_PENDING | bxxxxxx1 | A message associated with a transaction monitored by the MAF was delayed because the transaction had a GO pending in the requesting core. |
| VIC_PENDING | bxxxxxx1x | An LLC fill was delayed because the victimized data in the LLC was still being processed. |
| SNP_PENDING | bxxxxx1xx | A message associated with a transaction monitored by the MAF was delayed because the transaction had a snoop pending. |
| AC_PENDING | bxxxx1xxx | An incoming remote Intel QPI snoop was delayed because it conflicted with an existing MAF transaction that had an Ack Conflict pending. |
| IDX_BLOCK | bxxx1xxxx | An incoming local core RD that missed the LLC was delayed because a victim way could not be immediately chosen. |
| PA_BLOCK | bxx1xxxxx | If this count is very high, it likely means that software is frequently issuing requests to the same physical address from disparate threads simultaneously. Though there will also sometimes be a small number of PA_BLOCK nacks in the background due to cases when a pair of messages associated with the same transaction happen to arrive at the LLC at the same time and one of them gets delayed. |
| IDLE_QPI | bx1xxxxxx | Idle Intel QPI State |
| ALL_MAF_NACK2 | b1xxxxxxx | A message was rejected when one or more of the sub-events under MAF_NACK2 was true. This is included in MAF_NACK1 so that MAF_NACK1 with sub-event 0xFF will count the total number of Nacks. |
| TOTAL_MAF_NACKS | b11111111 | Total number of LLC pipeline passes that were nacked. |

MAF_NACK2

- **Title:** MAF NACK2
- **Category:** Local - MAF
- **Event Code:** 0x12, **Max. Inc/Cyc:** 1,
- **Definition:** Rejected (not-acknowledged) LLC pipeline passes (Set 2).

| Extension | umask [15:8] | Description |
|---------------------|--------------|---|
| --- | b00000000 | (*nothing will be counted*) |
| MAF_FULL | bxxxxxx1 | An incoming local processor RD/WR or remote Intel QPI snoop request that required a MAF entry was delayed because no MAF entry was available. |
| EGRESS_FULL | bxxxxxx1x | Some incoming message to the LLC that needed to generate a response message for transmission onto the ring was delayed due to ring back pressure. |
| VIQ_FULL | bxxxxx1xx | An incoming local processor RD request that missed the LLC was delayed because the LLC victim buffer was full. |
| NO_TRACKER_CREDITS | bxxxx1xxx | An incoming local processor RD or WR request was delayed because it required a Home tracker credit (for example, LLC RD Miss) and no credit was available. |
| NO_S_FIFO_CREDITS | bxxx1xxxx | Some incoming message to the LLC that needed to generate a message to the S-Box was delayed due to lack of available buffering resources in the S-Box. |
| NO_S_REQTBL_ENTRIES | bxx1xxxxx | An incoming local processor Rd or WR that needed to generate a transaction to Home (for example, LLC RD Miss) was delayed because the S-Box Request Table was full. |
| WB_PENDING | bx1xxxxxx | An incoming remote Intel QPI snoop request to the LLC was delayed because it conflicted with an existing transaction that had a WB to Home pending. |
| NACK2_ELSE | b1xxxxxxx | Some incoming message to the LLC was delayed for a reason not covered by any of the other MAF_NACK1 or MAF_NACK2 sub-events. |

OCCUPANCY_IPQ

- **Title:** IPQ Occupancy
- **Category:** Queue Occupancy
- **Event Code:** 0x1A, **Max. Inc/Cyc:** 8,
- **Definition:** Cumulative count of occupancy in the LLC's Ingress Probe Queue.

OCCUPANCY_IRQ

- **Title:** IRQ Occupancy
- **Category:** Queue Occupancy
- **Event Code:** 0x18, **Max. Inc/Cyc:** 24,
- **Definition:** Cumulative count of occupancy in the LLC's Ingress Response Queue.

OCCUPANCY_MAF

- **Title:** MAF Occupancy
- **Category:** Queue Occupancy
- **Event Code:** 0x1E, **Max. Inc/Cyc:** 16,
- **Definition:** Cumulative count of occupancy in the LLC's Miss Address File.

OCCUPANCY_RSPF

- **Title:** RSPF Occupancy
- **Category:** Queue Occupancy
- **Event Code:** 0x22, **Max. Inc/Cyc:** 8,
- **Definition:** Cumulative count of occupancy in the Snoop Response FIFO.

OCCUPANCY_RWRF

- **Title:** RWRF Occupancy
- **Category:** Queue Occupancy
- **Event Code:** 0x20, **Max. Inc/Cyc:** 12,
- **Definition:** Cumulative count of the occupancy in the Read/Write Request FIFO.

OCCUPANCY_VIQ

- **Title:** VIQ Occupancy
- **Category:** Queue Occupancy
- **Event Code:** 0x1C, **Max. Inc/Cyc:** 8,
- **Definition:** Cumulative count of the occupancy in the Victim Ingress Queue.

SINKS_C2P

- **Title:** C2P Sinks
- **Category:** Ring - WIR
- **Event Code:** 0x06, **Max. Inc/Cyc:** 3,
- **Definition:** Number of messages sunk by the processor that were sent by one of the C-Boxes.
- **NOTE:** Each sink represents the transfer of 32 bytes, or 2 sinks per cache line.

| Extension | umask [15:8] | Description |
|-----------|--------------|--|
| --- | b00000000 | (*nothing will be counted*) |
| IV | b00000001 | IV (C-Box snoops of a processor's cache) |
| AK | b00000010 | AK (GO messages send to the processor) |
| BL | b00000100 | BL (LLC data sent back to processor) |

SINKS_P2C

- **Title:** P2C Sinks
- **Category:** Ring - WIR
- **Event Code:** 0x05, **Max. Inc/Cyc:** 3,
- **Definition:** Number of messages sunk from the ring at the C-Box that were sent by one of the local processors.
- **NOTE:** Each sink represents the transfer of 32 bytes, or 2 sinks per cache line.

| Extension | umask [15:8] | Description |
|-----------|--------------|-------------------------------------|
| --- | b00000000 | (*nothing will be counted*) |
| AD | b00000001 | AD (Core RD/WR requests to the LLC) |

| Extension | umask [15:8] | Description |
|-----------|--------------|---|
| AK | b00000010 | AK (Core snoop responses to the LLC) |
| BL | b00000100 | BL (explicit and implicit WB data from the core to the LLC) |

SINKS_S2C

- **Title:** S2C Sinks
- **Category:** Ring - WIR
- **Event Code:** 0x07, **Max. Inc/Cyc:** 3,
- **Definition:** Number of messages sunk from the ring at the C-Box that were sent by the S-Box.
- **NOTE:** Each sink represents the transfer of 32 bytes, or 2 sinks per cache line.

| Extension | umask [15:8] | Description |
|-----------|--------------|--|
| --- | b00000000 | (*nothing will be counted*) |
| AD | b00000001 | AD (Intel QPI snoop request of LLC) |
| AK | b00000010 | AK (Intel QPI completions sent to LLC) |
| BL | b00000100 | BL (Data Fills sent to the LLC in response to RD requests) |

SINKS_S2P_BL

- **Title:** S2P Sinks
- **Category:** Ring - WIRa
- **Event Code:** 0x08, **Max. Inc/Cyc:** 1,
- **Definition:** Number BL ring messages sunk by the processor that were sent from the S-Box. This covers BL only, because that is the only kind of message the S-Box can send to a processor.
- **NOTE:** Each sink represents the transfer of 32 bytes, or 2 sinks per cache line.

SNP_HITS

- **Title:** Snoop Hits in LLC
- **Category:** Local - CC
- **Event Code:** 0x28, **Max. Inc/Cyc:** 1,
- **Definition:** Number of Intel QPI snoops that hit in the LLC according to state of LLC when the hit occurred. GotoS: LLC Data or Code Read Snoop Hit 'x' state in remote cache. GotoI: LLC Data Read for Ownership Snoop Hit 'x' state in remote cache.

| Extension | umask [15:8] | Description |
|-----------------|--------------|--|
| --- | b00000000 | (*nothing will be counted*) |
| REMOTE_RD_HITM | bxxxxxx1 | Intel QPI SnpData or SnpCode hit M line in LLC |
| REMOTE_RD_HITE | bxxxxx1x | Intel QPI SnpData or SnpCode hit E line in LLC |
| REMOTE_RD_HITS | bxxxx1xx | Intel QPI SnpData or SnpCode hit S line in LLC |
| REMOTE_RD_HITF | bxxxx1xxx | Intel QPI SnpData or SnpCode hit F line in LLC |
| REMOTE_RFO_HITM | bxxx1xxxx | Intel QPI SnpInvOwn or SnpInvtoE hit M line in LLC |
| REMOTE_RFO_HITE | bxx1xxxx | Intel QPI SnpInvOwn or SnpInvtoE hit E line in LLC |
| REMOTE_RFO_HITS | bx1xxxx | Intel QPI SnpInvOwn or SnpInvtoE hit S line in LLC |
| REMOTE_RFO_HITF | b1xxxxxx | Intel QPI SnpInvOwn or SnpInvtoE hit F line in LLC |
| REMOTE_HITM | bxx1xxx1 | Intel QPI Snoops that hit M line in LLC |
| REMOTE_HITE | bxx1xxx1x | Intel QPI Snoops that hit E line in LLC |

| Extension | umask [15:8] | Description |
|-------------|--------------|---|
| REMOTE_HITS | bx1xxx1xx | Intel QPI Snoops that hit S line in LLC |
| REMOTE_HITF | b1xxx1xxx | Intel QPI Snoops that hit F line in LLC |
| REMOTE_ANY | b11111111 | Intel QPI Snoops that hit in LLC (any line state) |

SNPS

- **Title:** Snoops to LLC
- **Category:** Local - CC
- **Event Code:** 0x27, **Max. Inc/Cyc:** 1,
- **Definition:** Number of Intel QPI snoops seen by the LLC.
- **NOTE:** Subtract CACHE_CHAR_QUAL.ANY_HIT from this event to determine how many snoops missed the LLC.

| Extension | umask [15:8] | Description |
|------------|--------------|--|
| --- | b00000000 | (*nothing will be counted*) |
| REMOTE_RD | b000000x1 | Remote Read - Goto S State. Intel QPI snoops (SnpData or SnpCode) to LLC that caused a transition to S in the cache. NOTE: ALL SnpData and SnpCode transactions are counted. If SnpData HITM policy is M->I, this subevent will capture those snoops. |
| REMOTE_RFO | b0000001x | Remote RFO - Goto I State. Intel QPI snoops (SnpInvOwn or SnpInvtoE) to LLC that caused an invalidate of a cache line. |
| REMOTE_ANY | b00000011 | Intel QPI snoops to LLC that hit in the cache line |

SPL_ARB_PRI_SW

- **Title:** Special ARB Priority Switch
- **Category:** EGL Special Request
- **Event Code:** 0x2A, **Max. Inc/Cyc:** 1,
- **Definition:** P2C Uncore/C6 SRAM request sunk when IRQ is full except dedicated entry free

SPL_CO_NSB

- **Title:** Special Request Collocated in NSB direction
- **Category:** EGL Special Request
- **Event Code:** 0x2D, **Max. Inc/Cyc:** 1,
- **Definition:** Uncore/SRAM request received from C-Box Egress in slot reserved for mine in NSB direction

SPL_CO_SB

- **Title:** Special Request Collocated in SB direction
- **Category:** EGL Special Request
- **Event Code:** 0x2C, **Max. Inc/Cyc:** 1,
- **Definition:** Uncore/SRAM request received from C-Box Egress in slot reserved for mine in SB direction

SPL_DEAD

- **Title:** Special Deadlock
- **Category:** EGL Special Request
- **Event Code:** 0x29, **Max. Inc/Cyc:** 1,
- **Definition:** P2C Uncore/C6 SRAM request sunk when ring request is bounced

SPL_EGR_NSB

- **Title:** Special Request at EGR in NSB direction
- **Category:** EGL Special Request
- **Event Code:** 0x30, **Max. Inc/Cyc:** 1,
- **Definition:** Special request at C-Box Egress output in NSB direction

SPL_EGR_SB

- **Title:** Special Request at EGR in SB direction
- **Category:** EGL Special Request
- **Event Code:** 0x2F, **Max. Inc/Cyc:** 1,
- **Definition:** Special request at C-Box Egress output in SB direction

SPL_IN_FULL_IRQ

- **Title:** Special Request Allocated in Full IRQ
- **Category:** EGL Special Request
- **Event Code:** 0x2E, **Max. Inc/Cyc:** 1,
- **Definition:** IRQ is full but special request allocated in IRQ when uncore count is 0

SPL_NOT_CO

- **Title:** Special Request Not Collocated
- **Category:** EGL Special Request
- **Event Code:** 0x2B, **Max. Inc/Cyc:** 1,
- **Definition:** Uncore/SRAM request sent but not co-located (illegal condition except for packet from Sbox)

SPOOF_ASSERT

- **Title:** IV Spoof Assertion Edge
- **Category:** EGL Miscellaneous
- **Event Code:** 0x3A, **Max. Inc/Cyc:** 1,
- **Definition:** IV spoof assertion edge

SPOOF_CRD_EMPTY

- **Title:** Spoof Credit Counter Empty
- **Category:** EGL Miscellaneous
- **Event Code:** 0x35, **Max. Inc/Cyc:** 1,
- **Definition:** Spoof credit counter is empty

SPOOF_DEASSERT

- **Title:** IV Spoof De-assertion Edge
- **Category:** EGL Miscellaneous
- **Event Code:** 0x3B, **Max. Inc/Cyc:** 1,
- **Definition:** IV spoof de-assertion edge

STARVED_EGRESS

- **Title:** Egress Queue Starved
- **Category:** Local - EGR
- **Event Code:** 0x0B, **Max. Inc/Cyc:** 8,
- **Definition:** Number of cycles that an Egress Queue is in starvation

| Extension | umask [15:8] | Description |
|-----------|--------------|---|
| --- | b00000000 | (*nothing will be counted*) |
| P2C_AD_SB | b00000001 | Processor-to-C-Box AD Egress that injects in the direction toward the nearest S-Box |
| C2S_AD_SB | b00000010 | C-Box-to-S-Box AD Egress. |
| AD_SB | b00000011 | Sum of AD Egresses that injects in the direction toward the nearest S-Box |
| AD_NSB | b00000100 | Sum across both AD Egress that inject in the direction away from the nearest S-Box. |
| AD | b00000111 | Sum across all AD Egresses |
| AK_SB | b00001000 | AK Egress that injects in the direction toward the nearest S-Box. |
| AK_NSB | b00010000 | AK Egress that injects in the direction away from the nearest S-Box. |
| AK | b00011000 | Sum across all AK Egresses. |
| BL_SB | b00100000 | BL Egress that injects in the direction toward the nearest S-Box. |
| BL_NSB | b01000000 | BL Egress that injects in the direction away from the nearest S-Box. |
| BL | b01100000 | Sum across all BL Egresses. |
| IV | b10000000 | IV Egress |

TRANS_IPQ

- **Title:** IPQ Transactions
- **Category:** Queue Occupancy
- **Event Code:** 0x1B, **Max. Inc/Cyc:** 1,
- **Definition:** Number of Intel QPI snoop probes that entered the LLC's Ingress Probe Queue.

TRANS_IRQ

- **Title:** IRQ Transactions
- **Category:** Queue Occupancy
- **Event Code:** 0x19, **Max. Inc/Cyc:** 1,
- **Definition:** Number of processor RD and/or WR requests to the LLC that entered the Ingress Response Queue.

TRANS_MAF

- **Title:** MAF Transactions
- **Category:** Queue Occupancy
- **Event Code:** 0x1F, **Max. Inc/Cyc:** 1,
- **Definition:** Number of transactions to allocate entries in LLC's Miss Address File.

TRANS_RSPF

- **Title:** RSPF Transactions
- **Category:** Queue Occupancy
- **Event Code:** 0x23, **Max. Inc/Cyc:** 1,
- **Definition:** Number of snoop responses from the processor that passed through the Snoop Response FIFO. The RSPF is a buffer that sits between each processor and the ring that buffers the processor's snoop responses in the event that there is back pressure due to ring congestion.

TRANS_RWRF

- **Title:** RWRF Transactions
- **Category:** Queue Occupancy
- **Event Code:** 0x21, **Max. Inc/Cyc:** 1,
- **Definition:** Number of requests that passed through the Read/Write Request FIFO. The RWRF is a buffer that sits between each processor and the ring that buffers the processor's RD/WR requests in the event that there is back pressure due to ring congestion.

TRANS_VIQ

- **Title:** VIQ Transactions
- **Category:** Queue Occupancy
- **Event Code:** 0x1D, **Max. Inc/Cyc:** 1,
- **Definition:** Number of LLC victims to enter the Victim Ingress Queue. All LLC victims pass through this queue. Including those that end up not requiring a WB.

2.4 B-Box Performance Monitoring

2.4.1 Overview of the B-Box

The B-Box is responsible for the protocol side of memory interactions, including coherent and non-coherent home agent protocols (as defined in the *Intel® QuickPath Interconnect Specification*). Additionally, the B-Box is responsible for ordering memory reads/writes to a given address such that the M-Box does not have to perform this conflict checking. All requests for memory attached to the coupled M-Box must first be ordered through the B-Box.

The B-Box has additional requirements on managing interactions with the M-Box, including RAS flows. All requests in-flight in the M-Box are tracked in the B-Box. The primary function of the B-Box, is as the coherent home agent for the Intel® QuickPath Interconnect cache coherence protocol. The home agent algorithm requires the B-Box to track outstanding requests, log snoop responses and other control messages, and make certain algorithmic decisions about how to respond to requests.

The B-Box only supports source snoopy Intel QuickPath Interconnect protocol flows.

2.4.2 B-Box Performance Monitoring Overview

Each of the two B-Boxes in the Intel® Xeon® Processor E7 Family supports event monitoring through four 48-bit wide counters (BBx_CR_B_MSR_PERF_CNT{3:0}). Each of these four counters is dedicated to observe a specific set of events as specified in its control register (BBx_CR_B_MSR_PERF_CTL{3:0}). The B-Box counters will increment by a maximum of 1 per cycle.

For information on how to setup a monitoring session, refer to [Section 2.1, "Global Performance Monitoring Control Counter Overflow"](#).

2.4.2.1 B-Box PMU - On Overflow and the Consequences (PMI/Freeze)

If an overflow is detected from a B-Box performance counter, the overflow bit is set at the box level (B_MSR_PMON_GLOBAL_STATUS.ov), and forwarded up the chain towards the U-Box. If a B-Box0 counter overflows, a notification is sent and stored in S-Box0 (S_MSR_PMON_SUMMARY.ov_mb) which, in turn, sends the overflow notification up to the U-Box (U_MSR_PMON_GLOBAL_STATUS.ov_s0). If a B-Box1 counter overflows, the overflow bit is set on the S-Box1 side of the hierarchy.

HW can be also configured (by setting the corresponding *.pmi_en* to 1) to send a PMI to the U-Box when an overflow is detected. The U-Box may be configured to freeze all uncore counting and/or send a PMI to selected cores when it receives this signal.

Once a freeze has occurred, in order to see a new freeze, the overflow field responsible for the freeze, found in B_MSR_PMON_GLOBAL_OVF_CTL.clr_ov, must be cleared. Assuming all the counters have been locally enabled (*.en* bit in data registers meant to monitor events) and the overflow bit(s) has been cleared, the B-Box is prepared for a new sample interval. Once the global controls have been re-enabled (Section 2.1.5, “Enabling a New Sample Interval from Frozen Counters”) counting will resume.

2.4.3 B-BOX Performance Monitors

Table 2-16. B-Box Performance Monitoring MSRs

| MSR Name | Access | MSR Address | Size (bits) | Description |
|----------------------------------|--------|-------------|-------------|--------------------------------------|
| BB1_CR_B_MSR_MASK | Rw_Rw | 0x0E4E | 64 | B-Box 1 PMON Mask Register |
| BB1_CR_B_MSR_MATCH | Rw_Rw | 0x0E4D | 64 | B-Box 1 PMON Match Register |
| BB0_CR_B_MSR_MASK | Rw_Rw | 0x0E46 | 64 | B-Box 0 PMON Mask Register |
| BB0_CR_B_MSR_MATCH | Rw_Rw | 0x0E45 | 64 | B-Box 0 PMON Match Register |
| | | | | |
| BB1_CR_B_MSR_PERF_CNT3_REG | Rw_Rw | 0x0C77 | 64 | B-Box 1 PMON Counter 3 |
| BB1_CR_B_MSR_PERF_CTL3_REG | Rw_Rw | 0x0C76 | 64 | B-Box 1 PMON Event Select 3 |
| BB1_CR_B_MSR_PERF_CNT2_REG | Rw_Rw | 0x0C75 | 64 | B-Box 1 PMON Counter 2 |
| BB1_CR_B_MSR_PERF_CTL2_REG | Rw_Rw | 0x0C74 | 64 | B-Box 1 PMON Event Select 2 |
| BB1_CR_B_MSR_PERF_CNT1_REG | Rw_Rw | 0x0C73 | 64 | B-Box 1 PMON Counter 1 |
| BB1_CR_B_MSR_PERF_CTL1_REG | Rw_Rw | 0x0C72 | 64 | B-Box 1 PMON Event Select 1 |
| BB1_CR_B_MSR_PERF_CNT0_REG | Rw_Rw | 0x0C71 | 64 | B-Box 1 PMON Counter 0 |
| BB1_CR_B_MSR_PERF_CTL0_REG | Rw_Rw | 0x0C70 | 64 | B-Box 1 PMON Event Select 0 |
| BB1_CR_C_MSR_PMON_GLOBAL_OVF_CTL | Rw_Rw | 0x0C62 | 32 | B-Box 1 PMON Global Overflow Control |
| BB1_CR_B_MSR_PMON_GLOBAL_STATUS | Rw_Rw | 0x0C61 | 32 | B-Box 1 PMON Global Status |
| BB1_CR_B_MSR_PERF_GLOBAL_CTL | Rw_Rw | 0x0C60 | 32 | B-Box 1 PMON Global Control |
| | | | | |
| BB0_CR_B_MSR_PERF_CNT3_REG | Rw_Rw | 0x0C37 | 64 | B-Box 0 PMON Counter 3 |
| BB0_CR_B_MSR_PERF_CTL3_REG | Rw_Rw | 0x0C36 | 64 | B-Box 0 PMON Event Select 3 |
| BB0_CR_B_MSR_PERF_CNT2_REG | Rw_Rw | 0x0C35 | 64 | B-Box 0 PMON Counter 2 |
| BB0_CR_B_MSR_PERF_CTL2_REG | Rw_Rw | 0x0C34 | 64 | B-Box 0 PMON Event Select 2 |
| BB0_CR_B_MSR_PERF_CNT1_REG | Rw_Rw | 0x0C33 | 64 | B-Box 0 PMON Counter 1 |
| BB0_CR_B_MSR_PERF_CTL1_REG | Rw_Rw | 0x0C32 | 64 | B-Box 0 PMON Event Select 1 |
| BB0_CR_B_MSR_PERF_CNT0_REG | Rw_Rw | 0x0C31 | 64 | B-Box 0 PMON Counter 0 |
| BB0_CR_B_MSR_PERF_CTL0_REG | Rw_Rw | 0x0C30 | 64 | B-Box 0 PMON Event Select 0 |

| MSR Name | Access | MSR Address | Size (bits) | Description |
|----------------------------------|--------|-------------|-------------|--------------------------------------|
| BBO_CR_C_MSR_PMON_GLOBAL_OVF_CTL | RW_RW | 0x0C22 | 32 | B-Box 0 PMON Global Overflow Control |
| BBO_CR_B_MSR_PMON_GLOBAL_STATUS | RW_RW | 0x0C21 | 32 | B-Box 0 PMON Global Status |
| BBO_CR_B_MSR_PERF_GLOBAL_CTL | RW_RW | 0x0C20 | 32 | B-Box 0 PMON Global Control |

2.4.3.1 B-Box Box Level PMON state

The following registers represent the state governing all box-level PMUs in the B-Box.

The `_GLOBAL_CTL` register contains the bits used to enable monitoring. It is necessary to set the `.ctr_en` bit to 1 before the corresponding data register can collect events.

If an overflow is detected from one of the B-Box PMON registers, the corresponding bit in the `_GLOBAL_STATUS.ov` field will be set. To reset the overflow bits set in the `_GLOBAL_STATUS.ov` field, a user must set the corresponding bits in the `_GLOBAL_OVF_CTL.clr_ov` field before beginning a new sample interval.

Table 2-17. B_MSR_PMON_GLOBAL_CTL Register – Field Definitions

| Field | Bits | Hw Reset Value | Description |
|---------------------|------|----------------|---|
| <code>ctr_en</code> | 3:0 | 0 | Must be set to enable each B-Box counter (bit 0 to enable <code>ctr0</code> , etc) NOTE: U-Box enable and per counter enable must also be set to fully enable the counter. |

Table 2-18. B_MSR_PMON_GLOBAL_STATUS Register – Field Definitions

| Field | Bits | Hw Reset Value | Description |
|-----------------|------|----------------|--|
| <code>ov</code> | 3:0 | 0 | If an overflow is detected from the corresponding B-Box PMON register, it's overflow bit will be set. NOTE: This bit is also cleared by setting the corresponding bit in <code>B_MSR_PMON_GLOBAL_OVF_CTL</code> |

Table 2-19. B_MSR_PMON_GLOBAL_OVF_CTL Register – Field Definitions

| Field | Bits | Hw Reset Value | Description |
|---------------------|------|----------------|--|
| <code>clr_ov</code> | 3:0 | 0 | Write '1' to reset the corresponding <code>B_MSR_PMON_GLOBAL_STATUS</code> overflow bit. |

2.4.3.2 B-Box PMON state - Counter/Control Pairs + Filters

The following table defines the layout of the B-Box performance monitor control registers. The main task of these configuration registers is to select the event to be monitored by their respective data counter. Setting the `.ev_sel` field performs the event selection. The `.en` bit which must be set to 1 to enable counting.

Additional control bits include:

- `.pmi_en` governs what to do if an overflow is detected.

NOTE: In the B-Box, each control register can only select from a specific set of events (see [Table 2-24, "Performance Monitor Events for B-Box Events"](#) for the mapping).

Table 2-20. B_MSR_PMON_EVT_SEL{3-0} Register – Field Definitions

| Field | Bits | HW Reset Value | Description |
|--------|-------|----------------|---|
| ig | 63 | 0 | Read zero; writes ignored. |
| rsv | 62:61 | 0 | Reserved; Must write to 0 else behavior is undefined. |
| ig | 60:51 | 0 | Read zero; writes ignored. |
| rsv | 50 | 0 | Reserved; Must write to 0 else behavior is undefined. |
| ig | 49:21 | 0 | Read zero; writes ignored. |
| pmi_en | 20 | 0 | When this bit is asserted and the corresponding counter overflows, a PMI exception is sent to the U-Box. |
| ig | 19:6 | 0 | Read zero; writes ignored. |
| ev_sel | 5:1 | 0 | Select event to be counted. NOTE: Event selects are NOT symmetric, each counter's event set is different. See event section and following tables for more details. |
| en | 0 | 0 | Local Counter Enable. When set, the associated counter is locally enabled. NOTE: It must also be enabled in B_MSR_PMON_GLOBAL_CTL and the U-Box to be fully enabled. |

The B-Box performance monitor data registers are 48b wide. A counter overflow occurs when a carry out bit from bit 47 is detected. Software can force all uncore counting to freeze after N events by preloading a monitor with a count value of $(2^{48} - 1) - N$ and setting the control register to send a PMI to the U-Box. Upon receipt of the PMI, the U-Box will disable counting ([Section 2.1.1, "Freezing on Counter Overflow"](#)). During the interval of time between overflow and global disable, the counter value will wrap and continue to collect events.

In this way, software can capture the precise number of events that occurred between the time uncore counting was enabled and when it was disabled (or 'frozen') with minimal skew.

If accessible, software can continuously read the data registers without disabling event collection.

Table 2-21. B_MSR_PMON_CNT{3-0} Register – Field Definitions

| Field | Bits | HW Reset Value | Description |
|-------------|------|----------------|----------------------------------|
| event_count | 47:0 | 0 | 48-bit performance event counter |

In addition to generic event counting, each B-Box provides a MATCH/MASK register pair that allow a user to filter packet traffic (incoming and outgoing) according to the packet Opcode, Message Class and Physical Address. Various events can be programmed to enable a B-Box performance counter (i.e. OPPOSITE_ADDR_IN_MATCH for counter 0) to capture the filter match as an event. The fields are laid out as follows:

Note: Refer to [Table 2-102, “Intel® QuickPath Interconnect Packet Message Classes”](#) and [Table 2-103, “Opcode Match by Message Class”](#) to determine the encodings of the B-Box Match Register fields.

Table 2-22. B_MSR_MATCH_REG Register – Field Definitions

| Field | Bits | HW Reset Value | Description |
|---------|-------|----------------|---|
| opc_out | 59:56 | 0 | Match to this outgoing opcode |
| opc_in | 55:52 | 0 | Match to this incoming opcode |
| msg_out | 51:48 | 0 | Match to this outgoing message class Message Class b0000 HOM - Requests b0001 HOM - Responses b0010 NDR b0011 SNP b0100 NCS --- b1100 NCB --- b1110 DRS |
| msg_in | 47:44 | 0 | Match to this incoming message class |
| addr | 43:0 | 0 | Match to this System Address - cache aligned address 49:6 |

Table 2-23. B_MSR_MASK_REG Register – Field Definitions

| Field | Bits | HW Reset Value | Description |
|---------|-------|----------------|---|
| opc_out | 59:56 | 0 | Mask for outgoing opcode |
| opc_in | 55:52 | 0 | Mask for incoming opcode |
| msg_out | 51:48 | 0 | Mask for outgoing message class |
| msg_in | 47:44 | 0 | Mask for incoming message class |
| addr | 43:0 | 0 | Mask this System Address - cache aligned address 49:6 |

2.4.4 B-Box Performance Monitoring Events

B-box PMUs allow users to monitor a number of latency related events. Traditionally, latency related events have been calculated by measuring the number of live transactions per cycle and accumulating this value in one of the PMU counters. The B-box offers a different approach. A number of occupancy

counters are dedicated to track live entries in different queues and structures. Rather than directly accumulate the occupancy values in the PMU counters, they are fed to a number of accumulators. Overflow of these accumulator values are then fed into the main PMU counters.

2.4.4.1 On the ARBQ:

The **ARBQ** (arbitration queue), used to store requests that are waiting for completion or arbitrating for resources, logically houses several smaller queues.

- **COHQ/1** (Coherence Queues) 256-entry - read request is pushed onto COHQ when it is in the 'ready' state (e.g. a read request that has received all of its snoop responses and is waiting for the M-Box ACK). All snoop responses received or RspFwd* or RspWb* will result in pushing that transaction onto the COHQ.
- **NDROQ** (NDR Output Queue) 256-entry - request is pushed when an NDR message has to be sent but is blocked due to lack of credits or the output port is busy.
- **SNPOQ** (SNP Output Queue) 256-entry - request is pushed when a snoop message has to be sent but is blocked due to lack of credits or the output port is busy.
- **DRSOQ** (DRS Output Queue) 32-entry - request is pushed when a DRS message has to be sent but is blocked due to lack of credits or the output port is busy.
- **MAQ** (M-Box Arbitration Queue) 32-entry - Request is pushed when it asks for M-Box access and the M-Box backpressures the B-Box (e.g. a link error flow in M-Box).
- **MDRSOQ** (Mirror DRS Output Queue) 32-entry - Request is pushed onto Mirror DRS output queue when a NonSnPWrData(Ptl) needs to be sent to the mirror slave and VN1 DRS channel or Intel QPI output resources are unavailable.
- **MHOMOQ** (Mirror HOM Output Queue) 256-entry - Request is pushed onto Mirror Home output queue when a Home message needs to be sent out from mirror master to mirror slave (NonSnPWr/RdCode/RdInvOwn) but is blocked due to a lack of credits (VN1 HOM) or the output port is busy.

2.4.4.2 On the Major B-Box Structures:

The 32-entry **IMT** (In Memory Table) Tracks and serializes in-flight reads and writes to the M-Box. The IMT ensures that there is only one pending request to a given system address. (NOTE: tracks all outstanding memory transactions that have already been sent to the M-Box.)

IMT average occupancy == (valid cnt * 32 / cycles)

IMT average latency == (valid cnt * 32 / IMT inserts)

The 256-entry **TF** (Tracker File) holds all transactions that arrive in the B-Box from the time they arrive until they are completed and leave the B-Box. Transactions could stay in this structure much longer than they are needed. IMT is the critical resource each transaction needs before being sent to the M-Box (memory controller)

TF average occupancy == (valid cnt * 256 / cycles)

TF average latency == (valid cnt * 256 / IMT inserts)

NOTE: The latency is only valid under 'normal' circumstances in which a request generates a memory prefetch. It will not hold true if the IMT is full and a prefetch isn't generated.

2.4.4.3 On InvItoE Transactions:

TF and IMT entries are broken into four categories: **writes** and **InvItoE** transactions (reads may be inferred by subtracting write and InvItoE transactions from all transactions) that do/do not come from **IOH** agents within the system. While reads and writes should be self-explanatory, InvItoE's may not be.

- **InvItOE** - returns ownership of line to requestor (in E-state) without returning data. The Home Agent sees it as an RFO and the memory controller sees it as memory read. The B-Box has to do more with it than it does for regular reads. It must make sure that not only the data gets forwarded to the requestor, but that it grants ownership to requestor. Depending on where the InvItOE request originates from, the HA takes different actions - if triggered by directory agent then HA/B-Box has to generate snoop invalidations to other caches/directories.

2.4.5 B-Box Events Ordered By Code

Table 2-24 summarizes the directly-measured B-Box events.

Table 2-24. Performance Monitor Events for B-Box Events

| Symbol Name | Event Code | Max Inc/ Cyc | Description |
|---------------------------------|------------|--------------|--------------------------------------|
| Counter 0 Events | | | |
| MSG_ADDR_IN_MATCH | 0x01 | 1 | Message + Address In Match |
| OPCODE_ADDR_IN_MATCH | 0x02 | 1 | Message + Opcode + Address In Match |
| MSG_OPCODE_ADDR_IN_MATCH | 0x03 | 1 | Opcode + Address In Match |
| TF_OCCUPANCY_ALL | 0x04 | 1 | TF Occupancy - All |
| TF_OCCUPANCY_WR | 0x05 | 1 | TF Occupancy - Writes |
| TF_OCCUPANCY_INVITOE | 0x06 | 1 | TF Occupancy - InvItOE |
| IMT_VALID_OCCUPANCY | 0x07 | 1 | IMT Valid Occupancy |
| DRSQ_OCCUPANCY | 0x09 | 1 | DRSQ Occupancy |
| TF_OCCUPANCY_IOH | 0x0B | 1 | TF Occupancy - All IOH |
| TF_OCCUPANCY_IOH_WR | 0x0D | 1 | TF Occupancy - IOH Writes |
| TF_OCCUPANCY_IOH_INVITOE | 0x0F | 1 | TF Occupancy - IOH InvItOE |
| SNPOQ_OCCUPANCY | 0x12 | 1 | SNPOQ Occupancy |
| DIRQ_OCCUPANCY | 0x17 | 1 | DIRQ Occupancy |
| TF_OCCUPANCY_IOH_NON_INVITOE_RD | 0x1C | 1 | TF Occupancy - IOH Non-InvItOE Reads |
| Counter 1 Events | | | |
| MSG_IN_MATCH | 0x01 | 1 | Message In Match |
| MSG_OUT_MATCH | 0x02 | 1 | Message Out Match |
| OPCODE_IN_MATCH | 0x03 | 1 | Opcode In Match |
| OPCODE_OUT_MATCH | 0x04 | 1 | Opcode Out Match |
| MSG_OPCODE_IN_MATCH | 0x05 | 1 | Message + Opcode In Match |
| MSG_OPCODE_OUT_MATCH | 0x06 | 1 | Message + Opcode Out Match |
| IMT_INSERTS_ALL | 0x07 | 1 | IMT All Inserts |
| DRSQ_INSERTS | 0x09 | 1 | DRSQ Inserts |
| IMT_INSERTS_IOH | 0x0A | 1 | IMT IOH Inserts |
| IMT_INSERTS_NON_IOH | 0x0B | 1 | IMT Non-IOH Inserts |
| IMT_INSERTS_WR | 0x0C | 1 | IMT Write Inserts |
| IMT_INSERTS_IOH_WR | 0x0D | 1 | IMT IOH Write Inserts |
| IMT_INSERTS_NON_IOH_WR | 0x0E | 1 | IMT Non-IOH Write Inserts |

Table 2-24. Performance Monitor Events for B-Box Events

| Symbol Name | Event Code | Max Inc/ Cyc | Description |
|-----------------------------|------------|-----------------|-----------------------------|
| IMT_INSERTS_INVITOE | 0x0F | 1 | IMT Invltoe Inserts |
| IMT_INSERTS_IOH_INVITOE | 0x10 | 1 | IMT IOH InvltoE Inserts |
| SNPOQ_INSERTS | 0x12 | 1 | SNPOQ Inserts |
| DIRQ_INSERTS | 0x17 | 1 | DIRQ Inserts |
| IMT_INSERTS_NON_IOH_INVITOE | 0x1C | 1 | IMT Non-IOH InvltoE Inserts |
| IMT_INSERTS_RD | 0x1D | 1 | IMT Read Inserts |
| IMT_INSERTS_NON_IOH_RD | 0x1F | 1 | IMT Non-IOH Read Inserts |
| Counter 2 Events | | | |
| MSGS_IN_NON_SNOOP | 0x01 | 1 | Incoming Non-Snoop Messages |
| MSGS_S_TO_B | 0x02 | 1 | SB Link (S to B) Messages |
| MSGS_B_TO_S | 0x03 | 1 | SB Link (B to S) Messages |
| ADDR_IN_MATCH | 0x04 | 1 | Address In Match |
| IMT_NE_CYCLES | 0x07 | 1 | IMT Non-Empty Cycles |
| Counter 3 Events | | | |
| EARLY_ACK | 0x02 | 1 | Early ACK |
| IMT_PREALLOC | 0x06 | 1 | IMT Prealloc |
| DEMAND_FETCH | 0x0F | 1 | Demand Fetches |
| IMPLICIT_WBS | 0x12 | 1 | Implicit WBS |
| COHQ_IMT_ALLOC_WAIT | 0x13 | 1 | COHQ IMT Allocation Wait |
| SBOX_VNO_UNAVAIL | 0x14 | 1 | S-Box VNO Unavailable |
| RBOX_VNA_UNAVAIL | 0x15 | 1 | R-Box VNA Unavailable |
| IMT_FULL | 0x16 | 1 | IMT Full |
| CONFLICTS | 0x17 | 1 | Conflicts |
| ACK_BEFORE_LAST_SNP | 0x19 | 1 | Ack Before Last Snoop |

2.4.6 B-Box Performance Monitor Event List

This section enumerates Intel® Xeon® Processor E7 Family uncore performance monitoring events for the B-Box.

ACK_BEFORE_LAST_SNP

- **Title:** Ack Before Last Snoop
- **Category:** Snoops
- **Event Code:** 0x19, **Max. Inc/Cyc:** 1, **PERF_CTL:** 3,
- **Definition:** Number of times M-Box acknowledge arrives before the last snoop response. For transactions issued to the memory controller (M-Box) as prefetches.

ADDR_IN_MATCH

- **Title:** Address In Match
- **Category:** Mask/Match
- **Event Code:** 0x04, **Max. Inc/Cyc:** 1, **PERF_CTL:** 2,
- **Definition:** Address Match at B-Box Input. Use B_MSR_MATCH/MASK_REG

CONFLICTS

- **Title:** Conflicts
- **Category:** Miscellaneous
- **Event Code:** 0x17, **Max. Inc/Cyc:** 1, **PERF_CTL:** 3,
- **Definition:** Number of conflicts.

COHQ_BYPASS

- **Title:** COHQ Bypass
- **Category:** ARB Queues
- **Event Code:** 0x0E, **Max. Inc/Cyc:** 1, **PERF_CTL:** 3,
- **Definition:** Coherence Queue Bypasses.

COHQ_IMT_ALLOC_WAIT

- **Title:** COHQ IMT Allocation Wait
- **Category:** ARB Queues
- **Event Code:** 0x13, **Max. Inc/Cyc:** 1, **PERF_CTL:** 3,
- **Definition:** Cycles Coherence Queue Waiting on IMT Allocation.

DIRQ_INSERTS

- **Title:** DIRQ Inserts
- **Category:** ARB Queues
- **Event Code:** 0x17, **Max. Inc/Cyc:** 1, **PERF_CTL:** 1,
- **Definition:** Directory Queue Inserts. Queue Depth is 256.

DIRQ_OCCUPANCY

- **Title:** DIRQ Occupancy
- **Category:** ARB Queues
- **Event Code:** 0x17, **Max. Inc/Cyc:** 1, **PERF_CTL:** 0,
- **Definition:** Directory Queue Occupancy. Queue Depth is 256.

DEMAND_FETCH

- **Title:** Demand Fetches
- **Category:** Miscellaneous
- **Event Code:** 0x0F, **Max. Inc/Cyc:** 1, **PERF_CTL:** 3,
- **Definition:** Counts number of times a memory access was issued after CohQ pop (i.e. IMT prefetch was not used).

DRSQ_INSERTS

- **Title:** DRSQ Inserts
- **Category:** ARB Queues
- **Event Code:** 0x09, **Max. Inc/Cyc:** 1, **PERF_CTL:** 1,
- **Definition:** DRSQ Inserts.

DRSQ_OCCUPANCY

- **Title:** DRSQ Occupancy
- **Category:** ARB Queues
- **Event Code:** 0x09, **Max. Inc/Cyc:** 1, **PERF_CTL:** 0,
- **Definition:** DRSQ Occupancy. Queue Depth is 4.

EARLY_ACK

- **Title:** Early ACK
- **Category:** Miscellaneous
- **Event Code:** 0x02, **Max. Inc/Cyc:** 1, **PERF_CTL:** 3,
- **Definition:** M-Box Early Acknowledgements.

IMPLICIT_WBS

- **Title:** Implicit WBS
- **Category:** Miscellaneous
- **Event Code:** 0x12, **Max. Inc/Cyc:** 1, **PERF_CTL:** 3,
- **Definition:** Number of Implicit Writebacks.

IMT_FULL

- **Title:** IMT Full
- **Category:** In-Flight Memory Table
- **Event Code:** 0x16, **Max. Inc/Cyc:** 1, **PERF_CTL:** 3,
- **Definition:** Number of times In-Flight Memory Table was full when entry was needed by incoming transaction.

IMT_INSERTS_ALL

- **Title:** IMT All Inserts
- **Category:** In-Flight Memory Table
- **Event Code:** 0x07, **Max. Inc/Cyc:** 1, **PERF_CTL:** 1,
- **Definition:** Inserts (all requests) to In-Flight Memory Table (e.g. all memory transactions targeting this B-Box as their home node and processed by this B-Box)
- **NOTE:** Conflicts and AckConflicts are considered IMT insert events. If the conflict rate ($\text{CONFLICTS} / \text{IMT_INSERTS_ALL} * 100$) is $> \sim 5\%$, it is not recommended that this event (along with `IMT_VALID_OCCUPANCY`) be used to derive average IMT latency or latency for specific flavors of inserts.

IMT_INSERTS_INVITOE

- **Title:** IMT InvItoE Inserts
- **Category:** In-Flight Memory Table
- **Event Code:** 0x0F, **Max. Inc/Cyc:** 1, **PERF_CTL:** 1,
- **Definition:** In-Flight Memory Table inserts of InvItoE requests (e.g. all InvItoE memory transactions targeting this B-Box as their home node and processed by this B-Box)
- **NOTE:** Conflicts and AckConflicts are considered IMT insert events. If the conflict rate ($\text{CONFLICTS} / \text{IMT_INSERTS_ALL} * 100$) is $> \sim 5\%$, it is not recommended that this event (along with `IMT_VALID_OCCUPANCY`) be used to derive average IMT latency or latency for specific flavors of inserts.

IMT_INSERTS_IOH

- **Title:** IMT IOH Inserts
- **Category:** In-Flight Memory Table
- **Event Code:** 0x0A, **Max. Inc/Cyc:** 1, **PERF_CTL:** 1,
- **Definition:** In-Flight Memory Table inserts of IOH requests (e.g. all IOH triggered memory transactions targeting this B-Box as their home node and processed by this B-Box)
- **NOTE:** Conflicts and AckConflicts are considered IMT insert events. If the conflict rate ($\text{CONFLICTS} / \text{IMT_INSERTS_ALL} * 100$) is $> \sim 5\%$, it is not recommended that this event (along with `IMT_VALID_OCCUPANCY`) be used to derive average IMT latency or latency for specific flavors of inserts.

IMT_INSERTS_IOH_INVITOE

- **Title:** IMT IOH InvItoE Inserts
- **Category:** In-Flight Memory Table
- **Event Code:** 0x10, **Max. Inc/Cyc:** 1, **PERF_CTL:** 1,
- **Definition:** In-Flight Memory Table inserts of IOH InvItoE requests (e.g. all IOH triggered InvItoE memory transactions targeting this B-Box as their home node and processed by this B-Box)
- **NOTE:** Conflicts and AckConflicts are considered IMT insert events. If the conflict rate ($\text{CONFLICTS}/\text{IMT_INSERTS_ALL} * 100$) is $> \sim 5\%$, it is not recommended that this event (along with $\text{IMT_VALID_OCCUPANCY}$) be used to derive average IMT latency or latency for specific flavors of inserts.

IMT_INSERTS_IOH_WR

- **Title:** IMT IOH Write Inserts
- **Category:** In-Flight Memory Table
- **Event Code:** 0x0D, **Max. Inc/Cyc:** 1, **PERF_CTL:** 1,
- **Definition:** In-Flight Memory Table Write IOH Request Inserts (e.g. all IOH triggered memory write transactions targeting this B-Box as their home node and processed by this B-Box)
- **NOTE:** Conflicts and AckConflicts are considered IMT insert events. If the conflict rate ($\text{CONFLICTS}/\text{IMT_INSERTS_ALL} * 100$) is $> \sim 5\%$, it is not recommended that this event (along with $\text{IMT_VALID_OCCUPANCY}$) be used to derive average IMT latency or latency for specific flavors of inserts.

IMT_INSERTS_NON_IOH

- **Title:** IMT Non-IOH Inserts
- **Category:** In-Flight Memory Table
- **Event Code:** 0x0B, **Max. Inc/Cyc:** 1, **PERF_CTL:** 1,
- **Definition:** In-Flight Memory Table inserts of Non-IOH requests (e.g. all non IOH triggered memory transactions targeting this B-Box as their home node and processed by this B-Box)
- **NOTE:** Conflicts and AckConflicts are considered IMT insert events. If the conflict rate ($\text{CONFLICTS}/\text{IMT_INSERTS_ALL} * 100$) is $> \sim 5\%$, it is not recommended that this event (along with $\text{IMT_VALID_OCCUPANCY}$) be used to derive average IMT latency or latency for specific flavors of inserts.

IMT_INSERTS_NON_IOH_INVITOE

- **Title:** IMT Non-IOH InvItoE Inserts
- **Category:** In-Flight Memory Table
- **Event Code:** 0x1C, **Max. Inc/Cyc:** 1, **PERF_CTL:** 1,
- **Definition:** In-Flight Memory Table inserts of Non-IOH InvItoE requests (e.g. all non IOH triggered InvItoE memory transactions targeting this B-Box as their home node and processed by this B-Box)
- **NOTE:** Conflicts and AckConflicts are considered IMT insert events. If the conflict rate ($\text{CONFLICTS}/\text{IMT_INSERTS_ALL} * 100$) is $> \sim 5\%$, it is not recommended that this event (along with $\text{IMT_VALID_OCCUPANCY}$) be used to derive average IMT latency or latency for specific flavors of inserts.

IMT_INSERTS_NON_IOH_RD

- **Title:** IMT Non-IOH Read Inserts
- **Category:** In-Flight Memory Table
- **Event Code:** 0x1F, **Max. Inc/Cyc:** 1, **PERF_CTL:** 1,
- **Definition:** In-Flight Memory Table inserts of Non-IOH read requests (e.g. all non IOH triggered memory read transactions targeting this B-Box as their home node and processed by this B-Box)
- **NOTE:** Conflicts and AckConflicts are considered IMT insert events. If the conflict rate ($\text{CONFLICTS}/\text{IMT_INSERTS_ALL} * 100$) is $> \sim 5\%$, it is not recommended that this event (along with $\text{IMT_VALID_OCCUPANCY}$) be used to derive average IMT latency or latency for specific flavors of inserts.

IMT_INSERTS_NON_IOH_WR

- **Title:** IMT Non-IOH Write Inserts
- **Category:** In-Flight Memory Table
- **Event Code:** 0x0E, **Max. Inc/Cyc:** 1, **PERF_CTL:** 1,
- **Definition:** In-Flight Memory Table Write Non-IOH Request Inserts (e.g. all non IOH triggered memory write transactions targeting this B-Box as their home node and processed by this B-Box)
- **NOTE:** Conflicts and AckConflicts are considered IMT insert events. If the conflict rate ($\text{CONFLICTS} / \text{IMT_INSERTS_ALL} * 100$) is $> \sim 5\%$, it is not recommended that this event (along with $\text{IMT_VALID_OCCUPANCY}$) be used to derive average IMT latency or latency for specific flavors of inserts.

IMT_INSERTS_RD

- **Title:** IMT Read Inserts
- **Category:** In-Flight Memory Table
- **Event Code:** 0x1D, **Max. Inc/Cyc:** 1, **PERF_CTL:** 1,
- **Definition:** In-Flight Memory Table inserts of read requests (e.g. all memory read transactions targeting this B-Box as their home node and processed by this B-Box)
- **NOTE:** Conflicts and AckConflicts are considered IMT insert events. If the conflict rate ($\text{CONFLICTS} / \text{IMT_INSERTS_ALL} * 100$) is $> \sim 5\%$, it is not recommended that this event (along with $\text{IMT_VALID_OCCUPANCY}$) be used to derive average IMT latency or latency for specific flavors of inserts.

IMT_INSERTS_WR

- **Title:** IMT Write Inserts
- **Category:** In-Flight Memory Table
- **Event Code:** 0x0C, **Max. Inc/Cyc:** 1, **PERF_CTL:** 1,
- **Definition:** In-Flight Memory Table Write Request Inserts (e.g. all memory write transactions targeting this B-Box as their home node and processed by this B-Box)
- **NOTE:** Conflicts and AckConflicts are considered IMT insert events. If the conflict rate ($\text{CONFLICTS} / \text{IMT_INSERTS_ALL} * 100$) is $> \sim 5\%$, it is not recommended that this event (along with $\text{IMT_VALID_OCCUPANCY}$) be used to derive average IMT latency or latency for specific flavors of inserts.

IMT_NE_CYCLES

- **Title:** IMT Non-Empty Cycles
- **Category:** In-Flight Memory Table
- **Event Code:** 0x07, **Max. Inc/Cyc:** 1, **PERF_CTL:** 2,
- **Definition:** In-Flight Memory Table Non-Empty Cycles.

IMT_PREALLOC

- **Title:** IMT Prealloc
- **Category:** In-Flight Memory Table
- **Event Code:** 0x06, **Max. Inc/Cyc:** 1, **PERF_CTL:** 3,
- **Definition:** In-Flight Memory Table in 1 DRS preallocation mode.

IMT_VALID_OCCUPANCY

- **Title:** IMT Valid Occupancy
- **Category:** In-Flight Memory Table
- **Event Code:** 0x07, **Max. Inc/Cyc:** 1, **PERF_CTL:** 0,
- **Definition:** In-Flight Memory Table (tracks memory transactions that have already been sent to the memory controller connected to this B-Box) valid occupancy. Indicates occupancy of the IMT.
- **NOTE:** A count of valid entries is accumulated every clock cycle in a subcounter. Since the IMT Queue Depth is 32, multiple this event by 32 to get a full count of valid IMT entries.

MSG_ADDR_IN_MATCH

- **Title:** Message + Address In Match
- **Category:** Mask/Match
- **Event Code:** 0x01, **Max. Inc/Cyc:** 1, **PERF_CTL:** 0,
- **Definition:** Message Class and Address Match at B-Box Input. Use B_MSR_MATCH/MASK_REG

MSG_B_TO_S

- **Title:** SB Link (B to S) Messages
- **Category:** S-Box Interface
- **Event Code:** 0x03, **Max. Inc/Cyc:** 1, **PERF_CTL:** 2,
- **Definition:** Number of SB Link (B to S) Messages (multiply by 9 to get flit count).

MSG_IN_MATCH

- **Title:** Message In Match
- **Category:** Mask/Match
- **Event Code:** 0x01, **Max. Inc/Cyc:** 1, **PERF_CTL:** 1,
- **Definition:** Message Class Match at B-Box Input. Use B_MSR_MATCH/MASK_REG

MSG_IN_NON_SNP

- **Title:** Incoming Non-Snoop Messages
- **Category:** Snoops
- **Event Code:** 0x01, **Max. Inc/Cyc:** 1, **PERF_CTL:** 2,
- **Definition:** Incoming Non-Snoop Messages.

MSG_OPCODE_ADDR_IN_MATCH

- **Title:** Message + Opcode + Address In Match
- **Category:** Mask/Match
- **Event Code:** 0x03, **Max. Inc/Cyc:** 1, **PERF_CTL:** 0,
- **Definition:** Message Class, Opcode and Address Match at B-Box Input. Use B_MSR_MATCH/MASK_REG

MSG_OPCODE_IN_MATCH

- **Title:** Message + Opcode In Match
- **Category:** Mask/Match
- **Event Code:** 0x05, **Max. Inc/Cyc:** 1, **PERF_CTL:** 1,
- **Definition:** Message Class and Opcode Match at B-Box Input. Use B_MSR_MATCH/MASK_REG

MSG_OPCODE_OUT_MATCH

- **Title:** Message + Opcode Out Match
- **Category:** Mask/Match
- **Event Code:** 0x06, **Max. Inc/Cyc:** 1, **PERF_CTL:** 1,
- **Definition:** Message Class and Opcode Match at B-Box Output. Use B_MSR_MATCH/MASK_REG

MSG_OUT_MATCH

- **Title:** Message Out Match
- **Category:** Mask/Match
- **Event Code:** 0x02, **Max. Inc/Cyc:** 1, **PERF_CTL:** 1,
- **Definition:** Message Class Match at B-Box Output. Use B_MSR_MATCH/MASK_REG

MSG_S_TO_B

- **Title:** SB Link (S to B) Messages
- **Category:** S-Box Interface
- **Event Code:** 0x02, **Max. Inc/Cyc:** 1, **PERF_CTL:** 2,
- **Definition:** Number of SB Link (S to B) Messages.

OPCODE_ADDR_IN_MATCH

- **Title:** Opcode + Address In Match
- **Category:** Mask/Match
- **Event Code:** 0x02, **Max. Inc/Cyc:** 1, **PERF_CTL:** 0,
- **Definition:** Opcode and Address Match at B-Box Input. Use B_MSR_MATCH/MASK_REG

OPCODE_IN_MATCH

- **Title:** Opcode In Match
- **Category:** Mask/Match
- **Event Code:** 0x03, **Max. Inc/Cyc:** 1, **PERF_CTL:** 1,
- **Definition:** Opcode Match at B-Box Input. Use B_MSR_MATCH/MASK_REG

OPCODE_OUT_MATCH

- **Title:** Opcode Out Match
- **Category:** Mask/Match
- **Event Code:** 0x04, **Max. Inc/Cyc:** 1, **PERF_CTL:** 1,
- **Definition:** Opcode Match at B-Box Output. Use B_MSR_MATCH/MASK_REG

RBOX_VNA_UNAVAIL

- **Title:** R-Box VNA Unavailable
- **Category:** R-Box Interface
- **Event Code:** 0x15, **Max. Inc/Cyc:** 1, **PERF_CTL:** 3,
- **Definition:** Number of times R-Box VNA credit was not available when needed.

SBOX_VN0_UNAVAIL

- **Title:** S-Box VN0 Unavailable
- **Category:** S-Box Interface
- **Event Code:** 0x14, **Max. Inc/Cyc:** 1, **PERF_CTL:** 3,
- **Definition:** Number of times S-Box VN0 credit was not available when needed.

SNPOQ_INSERTS

- **Title:** SNPOQ Inserts
- **Category:** ARB Queues
- **Event Code:** 0x12, **Max. Inc/Cyc:** 1, **PERF_CTL:** 1,
- **Definition:** SNP Output Queue Inserts. Queue Depth is 256.

SNPOQ_OCCUPANCY

- **Title:** SNPOQ Occupancy
- **Category:** ARB Queues
- **Event Code:** 0x12, **Max. Inc/Cyc:** 1, **PERF_CTL:** 0,
- **Definition:** SNP Output Queue Occupancy. Queue Depth is 256.

TF_ALL

- **Title:** TF Occupancy - All
- **Category:** Tracker File
- **Event Code:** 0x04, **Max. Inc/Cyc:** 1, **PERF_CTL:** 0,
- **Definition:** Tracker File occupancy for all requests. Accumulates lifetimes of all memory transactions that have arrived in this B-Box (TF starts tracking transactions before they are sent to the M-Box).
- **NOTE:** This event captures overflows from a subcounter tracking all requests. Multiply by 256 to determine the correct count.

TF_INVITOE

- **Title:** TF Occupancy - InvItoEs
- **Category:** Tracker File
- **Event Code:** 0x06, **Max. Inc/Cyc:** 1, **PERF_CTL:** 0,
- **Definition:** Tracker File occupancy for InvItoE requests. Accumulates lifetimes of InvItoE memory transactions that have arrived in this B-Box (TF starts tracking transactions before they are sent to the M-Box).
- **NOTE:** This event captures overflows from a subcounter tracking all requests. Multiply by 256 to determine the correct count.

TF_IOH

- **Title:** TF Occupancy - All IOH
- **Category:** Tracker File
- **Event Code:** 0x0B, **Max. Inc/Cyc:** 1, **PERF_CTL:** 0,
- **Definition:** Tracker File occupancy for IOH requests. Accumulates lifetimes of IOH triggered memory transactions that have arrived in this B-Box (TF starts tracking transactions before they are sent to the M-Box).
- **NOTE:** This event captures overflows from a subcounter tracking all requests. Multiply by 256 to determine the correct count.

TF_IOH_INVITOE

- **Title:** TF Occupancy - IOH InvItoEs
- **Category:** Tracker File
- **Event Code:** 0x0F, **Max. Inc/Cyc:** 1, **PERF_CTL:** 0,
- **Definition:** Tracker File occupancy for IOH InvItoE requests. Accumulates lifetimes of IOH triggered InvItoE memory transactions that have arrived in this B-Box (TF starts tracking transactions before they are sent to the M-Box).
- **NOTE:** This event captures overflows from a subcounter tracking all requests. Multiply by 256 to determine the correct count.

TF_IOH_NON_INVITOE_RD

- **Title:** TF Occupancy - IOH Non-InvItoE Reads
- **Category:** Tracker File
- **Event Code:** 0x1C, **Max. Inc/Cyc:** 1, **PERF_CTL:** 0,
- **Definition:** Tracker File occupancy for IOH Non-InvItoE read requests. Accumulates lifetimes of IOH triggered non-InvItoE memory transactions that have arrived in this B-Box (TF starts tracking transactions before they are sent to the M-Box).
- **NOTE:** This event captures overflows from a subcounter tracking all requests. Multiply by 256 to determine the correct count.

TF_IOH_WR

- **Title:** TF Occupancy - IOH Writes
- **Category:** Tracker File
- **Event Code:** 0x0D, **Max. Inc/Cyc:** 1, **PERF_CTL:** 0,
- **Definition:** Tracker File occupancy for IOH write requests. Accumulates lifetimes of IOH triggered write transactions that have arrived in this B-Box (TF starts tracking transactions before they are sent to the M-Box).
- **NOTE:** This event captures overflows from a subcounter tracking all requests. Multiply by 256 to determine the correct count.

TF_WR

- **Title:** TF Occupancy - Writes
- **Category:** Tracker File
- **Event Code:** 0x05, **Max. Inc/Cyc:** 1, **PERF_CTL:** 0,
- **Definition:** Tracker File occupancy for write requests. Accumulates lifetimes of write memory transactions that have arrived in this B-Box (TF starts tracking transactions before they are sent to the M-Box).
- **NOTE:** This event captures overflows from a subcounter tracking all requests. Multiply by 256 to determine the correct count.

2.5 S-Box Performance Monitoring

2.5.1 Overview of the S-Box

The S-Box represents the interface between the last level cache and the system interface. It manages flow control between the C and R & B-Boxes. The S-Box is broken into system bound (ring to Intel QPI) and ring bound (Intel QPI to ring) connections.

As such, it shares responsibility with the C-Box(es) as the Intel QPI caching agent(s). It is responsible for converting C-box requests to Intel QPI messages (i.e. snoop generation and data response messages from the snoop response) as well as converting/forwarding ring messages to Intel QPI packets and vice versa.

2.5.2 S-Box Performance Monitoring Overview

Each S-Box in the Intel® Xeon® Processor E7 Family supports event monitoring through 4 48b wide counters (S_MSR_PMON_CTR/CTL{3:0}). Each of these four counters can be programmed to count any S-Box event. the S-Box counters can increment by a maximum of 64 per cycle.

The S-Box also includes a mask/match register that allows a user to match packets leaving the S-Box according to various standard packet fields such as message class, opcode, etc. (NOTE: specifically goes with event 0, does not effect other events)

For information on how to setup a monitoring session, refer to [Section 2.1, "Global Performance Monitoring Control Counter Overflow"](#).

2.5.2.1 S-Box PMU - Overflow, Freeze and Unfreeze

If an overflow is detected from a S-Box performance counter, the overflow bit is set at the box level (S_MSR_PMON_GLOBAL_STATUS.ov), and forwarded up the chain to the U-Box where it will be stored in U_MSR_PMON_GLOBAL_STATUS.ov_s0. Each S-Box collects overflow bits for all boxes on it's 'side' of the chip. Refer to [Table 2-26, "S_MSR_PMON_SUMMARY Register Fields"](#) to determine how these bits are accumulated before they are forwarded to the U-Box.

HW can be also configured (by setting the corresponding *.pmi_en* to 1) to send a PMI to the U-Box when an overflow is detected. The U-Box may be configured to freeze all uncore counting and/or send a PMI to selected cores when it receives this signal.

Once a freeze has occurred, in order to see a new freeze, the overflow field responsible for the freeze, must be cleared by setting the corresponding bit in S_MSR_PMON_GLOBAL_OVF_CTL.clr_ov. Assuming all the counters have been locally enabled (.en bit in data registers meant to monitor events) and the overflow bit(s) has been cleared, the S-Box is prepared for a new sample interval. Once the global controls have been re-enabled (Section 2.1.5, “Enabling a New Sample Interval from Frozen Counters”), counting will resume.

Note: Due to the nature of the subcounters used in the S-Box, if a queue occupancy count event is set up to be captured, SW should set .reset_occ_cnt in the same write that the corresponding control register is enabled.

2.5.3 S-BOX Performance Monitors

Table 2-25. S-Box Performance Monitoring MSRs

| MSR Name | Access | MSR Address | Size (bits) | Description |
|---------------------------------|--------|-------------|-------------|--------------------------------------|
| SS1_CR_S_MSR_MASK | RW_RO | 0x0E5A | 64 | S-Box 1 Enable Mask Register |
| SS1_CR_S_MSR_MATCH | RW_RO | 0x0E59 | 64 | S-Box 1 Enable Match Register |
| SS1_CR_S_MSR_MM_CFG | RW_NA | 0x0E58 | 64 | S-Box 1 Enable Match/Mask Config |
| | | | | |
| SS0_CR_S_MSR_MASK | RW_RO | 0x0E4A | 64 | S-Box 1 Enable Mask Register |
| SS0_CR_S_MSR_MATCH | RW_RO | 0x0E49 | 64 | S-Box 1 Enable Match Register |
| SS0_CR_S_MSR_MM_CFG | RW_NA | 0x0E48 | 64 | S-Box 1 Enable Match/Mask Config |
| | | | | |
| SR1_CR_S_MSR_PMON_CTR3 | RW_RW | 0x0CD7 | 64 | S-Box 1 PMON Counter 3 |
| SR1_CR_S_MSR_PMON_CTL3 | RW_RO | 0x0CD6 | 64 | S-Box 1 PMON Control 3 |
| SR1_CR_S_MSR_PMON_CTR2 | RW_RW | 0x0CD5 | 64 | S-Box 1 PMON Counter 2 |
| SR1_CR_S_MSR_PMON_CTL2 | RW_RO | 0x0CD4 | 64 | S-Box 1 PMON Control 2 |
| SR1_CR_S_MSR_PMON_CTR1 | RW_RW | 0x0CD3 | 64 | S-Box 1 PMON Counter 1 |
| SR1_CR_S_MSR_PMON_CTL1 | RW_RO | 0x0CD2 | 64 | S-Box 1 PMON Control 1 |
| SR1_CR_S_MSR_PMON_CTR0 | RW_RW | 0x0CD1 | 64 | S-Box 1 PMON Counter 0 |
| SR1_CR_S_MSR_PMON_CTL0 | RW_RO | 0x0CD0 | 64 | S-Box 1 PMON Control 0 |
| | | | | |
| SR1_CR_S_MSR_PMON_SUMMARY | RO_WO | 0x0CC3 | 32 | S-Box 1 PMON Global Summary |
| SR1_CR_S_MSR_PMON_OVF_CTL | WO_RO | 0x0CC2 | 32 | S-Box 1 PMON Global Overflow Control |
| SR1_CR_S_MSR_PMON_GLOBAL_STATUS | RW_RW | 0x0CC1 | 32 | S-Box 1 PMON Global Overflow Status |
| SR1_CR_S_MSR_PMON_GLOBAL_CTL | RW_RO | 0x0CC0 | 32 | S-Box 1 PMON Global Control |
| | | | | |
| SR0_CR_S_MSR_PMON_CTR3 | RW_RW | 0x0C57 | 64 | S-Box 0 PMON Counter 3 |
| SR0_CR_S_MSR_PMON_CTL3 | RW_RO | 0x0C56 | 64 | S-Box 0 PMON Control 3 |
| SR0_CR_S_MSR_PMON_CTR2 | RW_RW | 0x0C55 | 64 | S-Box 0 PMON Counter 2 |
| SR0_CR_S_MSR_PMON_CTL2 | RW_RO | 0x0C54 | 64 | S-Box 0 PMON Control 2 |
| SR0_CR_S_MSR_PMON_CTR1 | RW_RW | 0x0C53 | 64 | S-Box 0 PMON Counter 1 |
| SR0_CR_S_MSR_PMON_CTL1 | RW_RO | 0x0C52 | 64 | S-Box 0 PMON Control 1 |
| SR0_CR_S_MSR_PMON_CTR0 | RW_RW | 0x0C51 | 64 | S-Box 0 PMON Counter 0 |
| SR0_CR_S_MSR_PMON_CTL0 | RW_RO | 0x0C50 | 64 | S-Box 0 PMON Control 0 |

| MSR Name | Access | MSR Address | Size (bits) | Description |
|---------------------------------|--------|-------------|-------------|--------------------------------------|
| SRO_CR_S_MSR_PMON_SUMMARY | RO_WO | 0x0C43 | 32 | S-Box 0 PMON Global Summary |
| SRO_CR_S_MSR_PMON_OVF_CTL | WO_RO | 0x0C42 | 32 | S-Box 0 PMON Global Overflow Control |
| SRO_CR_S_MSR_PMON_GLOBAL_STATUS | RW_RW | 0x0C41 | 32 | S-Box 0 PMON Global Overflow Status |
| SRO_CR_S_MSR_PMON_GLOBAL_CTL | RW_RO | 0x0C40 | 32 | S-Box 0 PMON Global Control |

2.5.3.1 S-Box PMON for Global State

The S_MSR_PMON_SUMMARY in each S-Box collects overflow bits from the boxes attached to it and forwards them to the U-Box.

Table 2-26. S_MSR_PMON_SUMMARY Register Fields

| Field | Bits | HW Reset Value | Description |
|--------|-------|----------------|---|
| ig | 63:21 | | Read zero; writes ignored. |
| ov_c_h | 20 | 0 | Overflow in 'hi' C Box In SBOX0, indicates overflow in C-Box 4. In SBOX1, indicates overflow in C-Box 9. |
| ov_r | 19 | 0 | Overflow in R Box In S-Box0, indicates overflow from Left R-Box In S-Box1, indicates overflow from Right R-Box |
| ov_s | 18 | 0 | Overflow in S Box |
| ig | 17 | | Read zero; writes ignored. |
| ov_mb | 16 | 0 | Overflow in M- or B-Box |
| ig | 15:3 | | Read zero; writes ignored. |
| ov_c_m | 2 | 0 | Overflow in 'mid' C-Boxes In SBOX0, indicates overflow in C-Box 2 or 3. In SBOX1, indicates overflow in C-Box 7 or 8. |
| ig | 1 | | Read zero; writes ignored. |
| ov_c_l | 0 | 0 | Overflow in 'low' C-Boxes In SBOX0, indicates overflow in C-Box 0 or 1. In SBOX1, indicates overflow in C-Box 5 or 6. |

2.5.3.2 S-Box Box Level PMON state

The following registers represent the state governing all box-level PMUs in the S-Box.

The _GLOBAL_CTL register contains the bits used to enable monitoring. It is necessary to set the .ctr_en bit to 1 before the corresponding data register can collect events.

If an overflow is detected from one of the S-Box PMON registers, the corresponding bit in the _GLOBAL_STATUS.ov field will be set. To reset the overflow bits set in the _GLOBAL_STATUS.ov field, a user must set the corresponding bits in the _GLOBAL_OVF_CTL.clr_ov field before beginning a new sample interval.

Table 2-27. S_MSR_PMON_GLOBAL_CTL Register Fields

| Field | Bits | HW Reset Value | Description |
|--------|------|----------------|--|
| ctr_en | 3:0 | 0 | Must be set to enable each SBOX counter (bit 0 to enable ctr0, etc) NOTE: U-Box enable and per counter enable must also be set to fully enable the counter. |

Table 2-28. S_MSR_PMON_GLOBAL_STATUS Register Fields

| Field | Bits | HW Reset Value | Description |
|-------|------|----------------|--|
| ov | 3:0 | 0 | If an overflow is detected from the corresponding SBOX PMON register, it's overflow bit will be set. |

Table 2-29. S_MSR_PMON_OVF_CTRL Register Fields

| Field | Bits | HW Reset Value | Description |
|--------|------|----------------|---|
| clr_ov | 3:0 | 0 | Writing '1' to bit in field causes corresponding bit in 'Overflow PerfMon Counter' field in S_CSR_PMON_GLOBAL_STATUS register to be cleared to 0. |

2.5.3.3 S-Box PMON state - Counter/Control Pairs + Filters

The following table defines the layout of the S-Box performance monitor control registers. The main task of these configuration registers is to select the event to be monitored by their respective data counter. Setting the *.ev_sel* field performs the event selection. The *.en* bit must be set to 1 to enable counting.

Additional control bits include:

- *.pmi_en* governs what to do if an overflow is detected.
- *.threshold* - If the *.threshold* is set to a non-zero value, that value is compared against the incoming count for that event in each cycle. If the incoming count is \geq the threshold value, then the event count captured in the data register will be incremented by 1.
- *.invert* - Changes the *.threshold* test condition to '<'
- *.edge_detect* - Rather than accumulating the raw count each cycle (for events that can increment by 1 per cycle), the register can capture transitions from no event to an event incoming.
- *.reset_occ_cnt* - Reset 7b occupancy counter associated with this counter.

Table 2-30. S_CSR_PMON_CTL{3-0} Register – Field Definitions

| Field | Bits | HW Reset Value | Description |
|---------------|-------|----------------|---|
| ig | 63 | 0 | Read zero; writes ignored. |
| rsv | 62:61 | 0 | Reserved; Must write to 0 else behavior is undefined. |
| ig | 60:32 | 0 | Read zero; writes ignored. |
| threshold | 31:24 | 0 | Threshold used for counter comparison. |
| invert | 23 | 0 | Invert threshold comparison. When '0', the comparison will be thresh >= event. When '1', the comparison will be threshold < event. |
| en | 22 | 0 | Local Counter Enable. When set, the associated counter is locally enabled. NOTE: It must also be enabled in S_MSR_PMON_GLOBAL_CTL and the U-Box to be fully enabled. |
| ig | 21 | 0 | Read zero; writes ignored. |
| pmi_en | 20 | 0 | PMI Enable. If bit is set, when corresponding counter overflows, a PMI exception is sent to the U-Box.w |
| ig | 19 | 0 | Read zero; writes ignored. |
| edge_detect | 18 | 0 | Edge Detect. When bit is set, 0->1 transition of a one bit event input will cause counter to increment. When bit is 0, counter will increment for however long event is asserted. |
| reset_occ_cnt | 17 | 0 | Reset Occupancy Counter associated with this counter. |
| ig | 16 | 0 | Read zero; writes ignored. |
| umask | 15:8 | 0 | Unit Mask - select subevent of event. |
| ev_sel | 7:0 | 0 | Event Select |

The S-Box performance monitor data registers are 48b wide. A counter overflow occurs when a carry out bit from bit 47 is detected. Software can force all uncore counting to freeze after N events by preloading a monitor with a count value of $(2^{48} - 1) - N$ and setting the control register to send a PMI to the U-Box. Upon receipt of the PMI, the U-Box will disable counting ([Section 2.1.1, “Freezing on Counter Overflow”](#)). During the interval of time between overflow and global disable, the counter value will wrap and continue to collect events.

In this way, software can capture the precise number of events that occurred between the time uncore counting was enabled and when it was disabled (or ‘frozen’) with minimal skew.

If accessible, software can continuously read the data registers without disabling event collection.

Table 2-31. S_CSR_PMON_CTR{3-0} Register – Field Definitions

| Field | Bits | HW Reset Value | Description |
|-------------|------|----------------|----------------------------------|
| event_count | 47:0 | 0 | 48-bit performance event counter |

2.5.3.4 S-Box Registers for Mask/Match Facility

In addition to generic event counting, the uncore provides a MATCH/MASK register pair for each S-Box that allows a user to filter outgoing packet traffic (system bound) according to the packet Opcode, Message Class, Response, HNID and Physical Address. Program the selected S-Box counter to capture TO_R_PROG_EV to capture the filter match as an event.

To use the match/mask facility :

a) Set MM_CFG (see Table 2-32, “S_MSR_MM_CFG Register – Field Definitions”) reg bit 63 to 0.

b) Program match/mask regs (see Table 2-33, “S_MSR_MATCH Register – Field Definitions”). (if MM_CFG[63] == 1, a write to match/mask will produce a GP fault).

NOTE: The address and the Home Node ID have a mask component in the MASK register. To mask off other fields (e.g. opcode or message class), set the field to all 0s.

c) Set the counter’s control register event select to 0x0 (TO_R_PROG_EV) to capture the mask/match as a performance event.

d) Set MM_CFG reg bit 63 to 1 to start matching.

Table 2-32. S_MSR_MM_CFG Register – Field Definitions

| Field | Bits | HW Reset Value | Description |
|-------|------|----------------|----------------------------|
| mm_en | 63 | 0 | Enable Match/Mask |
| ig | 62:0 | | Read zero; writes ignored. |

Table 2-33. S_MSR_MATCH Register – Field Definitions

| Field | Bits | HW Reset Value | Description |
|-------|-------|----------------|--|
| resp | 63:59 | 0 | Match if returning data is in b1xxxx - 'F' state. bx1xxx - 'S' state bxx1xx - 'E' state. bxxx1x - 'M' state. bxxxx1 - 'I' state. NOTE: Response matching is only done on the DRS response to a HOMO RdCode, RdData or RdInvOwn request. NOTE: Match will be ignored if field set to all 0s. |
| opc | 58:48 | 0 | Match on Opcode (see Table 2-34, “S_MSR_MATCH.opc - Opcode Match by Message Class”) NOTE: Match will be ignored if field set to all 0s. |
| mc | 47:43 | 0 | Match on Message Class b1xxxx - NCB bx1xxx - NCS bxx1xx - NDR bxxx1x - HOM1 bxxxx1 - HOM0 NOTE: Match will be ignored if field set to all 0s. |
| addr | 42:5 | 0 | Match on PA address bits [43:6] |
| hnid | 4:0 | 0 | Match on Home NodeID |

Refer to Table 2-104, “Opcodes (Alphabetical Listing)” for definitions of the opcodes found in the following table.

Table 2-34. S_MSR_MATCH.opc - Opcode Match by Message Class

| bit | NCB | NCS | NDR | HOM1 | HOMO |
|-----|-------------|---------|------|-----------|-------------|
| 58 | --- | NcMsgS | --- | --- | --- |
| 57 | --- | NcIOWr | --- | --- | --- |
| 56 | DebugData | --- | --- | RspSWb | --- |
| 55 | WcWrPtl | NcCfgWr | --- | RsplWb | AckCnflt |
| 54 | NcWrPtl | NcIORd | --- | RspFwdSwb | WbMtoE |
| 53 | IntPriorUpd | --- | --- | RspFwdlWb | WbMtol |
| 52 | IntPhysical | NcCfgRd | --- | RspFwdS | AckCnfltWbl |
| 51 | IntLogical | NcRdPtl | --- | RspFwdl | InvltoE |
| 50 | NcMsgB | FERR | --- | RspCnflt | RdInvOwn |
| 49 | WcWr | IntAck | CmpD | RspS | RdData |
| 48 | NcWr | NcRd | Cmp | Rspl | RdCode |

Table 2-35. S_MSR_MASK Register – Field Definitions

| Field | Bits | HW Reset Value | Description |
|-------|-------|----------------|---|
| ig | 62:39 | | Read zero; writes ignored. |
| addr | 38:1 | 0 | Mask PA address bits [43:6]. For each mask bit that is set, the corresponding bit in the address is already considered matched (e.g. it is ignored). If it is clear the it must match the corresponding address match bit in the S_MSR_MATCH register. |
| hnid | 0 | 0 | Disable HNID matching. 1 - HNID is NOT matched 0 - HNID is compared against the match |

2.5.4 S-BOX Performance Monitoring Events

2.5.4.1 An Overview

The S-Box provides events to track incoming (ring bound)/outgoing (system bound) transactions, various queue occupancies/latencies that track those transactions and a variety of static events such as bypass use (i.e. EGRESS_BYPASS) and when output credit is unavailable (i.e. NO_CREDIT_SNP). Many of these events can be further broken down by message class.

2.5.4.2 On Queue Occupancy Usage

This means two things:

- none of the physical queues receive more than one entry per cycle
- The entire 7b from the ‘selected’ (by the event select) queue occ subcounter is sent to the generic counter each cycle, meaning that the max inc of a generic counter is 64 (for the sys bound HOM buffer).

Associated with each of the four general purpose counters is a 7b queue occupancy counter which supports the various queue occupancy events found in [Section 2.5.5, "S-Box Events Ordered By Code"](#).

Each System Bound and Ring Bound data storage structure within the S-Box (queue/FIFO/buffer) has an associated tally counter which can be used to provide input into one of the S-Box performance counters. The data structure the tally counter is 'attached' to then sends increment/decrement signals as it receives/removes new entries. The tally counter then sends its contents to the performance counter each cycle.

The following table summarizes the queues (and their associated events) responsible for buffering traffic into and out of the S-Box.

Table 2-36. S-Box Data Structure Occupancy Events

| Structure/Event Name | Subev | Max Entries | Instances | Description/Comment |
|---|-------|-------------|-----------|--|
| System Bound HOM Message Queue TO_R_B_HOM_MSGQ_OCCUPANCY | RBOX | 64 | 2 | HOM Packet to System 1 buffer for R-Box and 1 for B-Box, 64 entries each. The sum of the occupied entries in the two header buffers will never exceed 64. |
| | B-Box | 64 | | |
| | ALL | 64 | | |
| System Bound SNP Message Queue TO_R_SNP_MSGQ_OCCUPANCY | | 32 | 1 | SNP Packet to System |
| System Bound NDR Message Queue TO_R_NDR_MSGQ_OCCUPANCY | | 16 | 1 | NDR Packet to System |
| System Bound DRS Message Queue TO_R_DRS_MSGQ_OCCUPANCY | | 16 | 4 | DRS Packet to System |
| System Bound NCB Message Queue TO_R_NCB_MSGQ_OCCUPANCY | | 16 | 4 | NCB Packet to System |
| System Bound NCS Message Queue TO_R_NCS_MSGQ_OCCUPANCY | | 16 | 4 | NCS Packet to System |
| Ring Bound Message Queue TO_RING_MSGQ_OCCUPANCY | SNP | 31 | 1 | Packets from System (SNP/NCS/NCB - NDR is separate) The total of the buffer entries occupied by all 3 message classes will never exceed 36. |
| | NCS | 4 | | |
| | NCB | 4 | | |
| | ALL | 36 | | |
| Ring Bound NDR Message Queue TO_RING_NDR_MSGQ_OCCUPANCY | | 32 | 1 | NDR Packet from System |
| Ring Bound R2S Message Queue TO_RING_R2S_MSGQ_OCCUPANCY | | 8 | 1 | DRS Packet from R-Box |
| Ring Bound B2S Message Queue TO_RING_B2S_MSGQ_OCCUPANCY | | 8 | 1 | DRS Packet from B-Box |
| Request Table | | 48 | 1 | System Bound Request |

2.5.4.3 On Packet Transmission Events

For the message classes that have variable length messages, the S-Box has separate events which count the number of flits of those message classes sent or received (i.e PKTS_SENT_HOM vs. PKTS/FLITS_SENT_NCB). For message classes that have fixed length messages, the total number of flits can be calculated by multiplying the total messages by the number of flits per message (i.e. PKTS_RCVD_NCB).

| Message Class | Flits per Msg | Comment |
|----------------|---------------|--|
| HOM | 1 | |
| SNP | 1 | |
| NDR | 1 | |
| Ring Bound DRS | 9 | R2S and B2S DRS messages are always full cacheline messages which are 9 flits. NOTE: flits are variable in the Sys Bound direction. |
| Ring Bound NCS | 3 | The only ring bound NCS message type is NcMsgS. There are always 3 flits. NOTE: flits are variable in the Sys Bound direction. |
| Ring Bound NCB | 11 | The only ring bound NCB message types are: NcMsgB, IntLogical, IntPhysical. These are all 11 flit messages. NOTE: flits are variable in the Sys Bound direction. |

The number of flits sent or received can be divided by the total number of uncore cycles (see [Section 2.8.2, “W-Box Performance Monitoring Overview”](#)) to calculate the link utilization for each message class. The combined number of flits across message classes can be used to calculate the total link utilization.

Note that for S2R and R2S links, there is no single event which counts the total number of message and credit carrying idle flits sent on the link. The total link utilization can be approximated by adding together the number of flits of the message classes that are expected to be most frequent.

2.5.5 S-Box Events Ordered By Code

Table 2-37 summarizes the directly-measured S-Box events.

Table 2-37. Performance Monitor Events for S-Box Events

| Symbol Name | Event Code | Max Inc/ Cyc | Description |
|-----------------------------|------------|--------------|--|
| TO_R_PROG_EV | 0x00 | 1 | System Bound Programmable Event |
| TO_R_B_HOM_MSGQ_CYCLES_FULL | 0x03 | 1 | Cycles System Bound HOM Message Queue Full. |
| TO_R_B_HOM_MSGQ_CYCLES_NE | 0x06 | 1 | Cycles System Bound HOM Message Queue Not Empty. |
| TO_R_B_HOM_MSGQ_OCCUPANCY | 0x07 | 64 | System Bound HOM Message Queue Occupancy |
| TO_R_SNP_MSGQ_CYCLES_FULL | 0x08 | 1 | Cycles System Bound SNP Message Queue Full |
| TO_R_SNP_MSGQ_CYCLES_NE | 0x09 | 1 | Cycles System Bound SNP Message Queue Not Empty |
| TO_R_SNP_MSGQ_OCCUPANCY | 0x0A | 32 | System Bound SNP Message Queue Occupancy |
| TO_R_NDR_MSGQ_CYCLES_FULL | 0x0B | 1 | Cycles System Bound NDR Message Queue Full. |
| TO_R_NDR_MSGQ_CYCLES_NE | 0x0C | 1 | Cycles System Bound NDR Message Queue Not Empty |
| TO_R_NDR_MSGQ_OCCUPANCY | 0x0D | 16 | System Bound NDR Message Queue Occupancy |
| TO_R_DRS_MSGQ_CYCLES_FULL | 0x0E | 1 | Cycles System Bound DRS Message Queue Full |
| TO_R_DRS_MSGQ_CYCLES_NE | 0x0F | 1 | Cycles System Bound DRS Message Queue Not Empty |
| TO_R_DRS_MSGQ_OCCUPANCY | 0x10 | 64 | System Bound DRS Message Queue Occupancy |
| TO_R_NCB_MSGQ_CYCLES_FULL | 0x11 | 1 | Cycles System Bound NCB Message Queue Full |

Table 2-37. Performance Monitor Events for S-Box Events

| Symbol Name | Event Code | Max Inc/ Cyc | Description |
|------------------------------|------------|--------------|---|
| TO_R_NCB_MSGQ_CYCLES_NE | 0x12 | 1 | Cycles System Bound NCB Message Queue Not Empty |
| TO_R_NCB_MSGQ_OCCUPANCY | 0x13 | 64 | System Bound NCB Message Queue Occupancy |
| TO_R_NCS_MSGQ_CYCLES_FULL | 0x14 | 1 | Cycles System Bound NCS Message Queue Full |
| TO_R_NCS_MSGQ_CYCLES_NE | 0x15 | 1 | Cycles System Bound NCS Message Queue Not Empty |
| TO_R_NCS_MSGQ_OCCUPANCY | 0x16 | 64 | System Bound NCS Message Queue Occupancy |
| TO_RING_SNP_MSGQ_CYCLES_FULL | 0x20 | 1 | Cycles Ring Bound SNP Message Queue Full |
| TO_RING_NCB_MSGQ_CYCLES_FULL | 0x21 | 1 | Cycles Ring Bound NCB Message Queue Full |
| TO_RING_NCS_MSGQ_CYCLES_FULL | 0x22 | 1 | Cycles Ring Bound NCS Message Queue Full |
| TO_RING_SNP_MSGQ_CYCLES_NE | 0x23 | 1 | Cycles Ring Bound SNP Message Queue Not Empty |
| TO_RING_NCB_MSGQ_CYCLES_NE | 0x24 | 1 | Cycles Ring Bound NCB Message Queue Not Empty |
| TO_RING_NCS_MSGQ_CYCLES_NE | 0x25 | 1 | Cycles Ring Bound NCS Message Queue Not Empty |
| TO_RING_MSGQ_OCCUPANCY | 0x26 | 361 | Cycles Ring Bound Message Queue Occupancy |
| TO_RING_NDR_MSGQ_CYCLES_FULL | 0x27 | 1 | Cycles Ring Bound NDR Message Queue Full. |
| TO_RING_NDR_MSGQ_CYCLES_NE | 0x28 | 1 | Cycles Ring Bound NDR Message Queue Not Empty |
| TO_RING_NDR_MSGQ_OCCUPANCY | 0x29 | 321 | Ring Bound NDR Message Queue Occupancy |
| TO_RING_R2S_MSGQ_CYCLES_FULL | 0x2A | 1 | Cycles Ring Bound R2S Message Queue Full. |
| TO_RING_R2S_MSGQ_CYCLES_NE | 0x2B | 1 | Cycles Ring Bound R2S Message Queue Not Empty |
| TO_RING_R2S_MSGQ_OCCUPANCY | 0x2C | 8 | Ring Bound R2S Message Queue Occupancy |
| TO_RING_B2S_MSGQ_CYCLES_FULL | 0x2D | 1 | Cycles Ring Bound B2S Message Queue Full. |
| TO_RING_B2S_MSGQ_CYCLES_NE | 0x2E | 1 | Cycles Ring Bound B2S Message Queue Not Empty |
| TO_RING_B2S_MSGQ_OCCUPANCY | 0x2F | 8 | Ring Bound B2S Message Queue Occupancy |
| HALFLINE_BYPASS | 0x30 | 1 | Half Cacheline Bypass |
| REQ_TBL_OCCUPANCY | 0x31 | 48 | Request Table Occupancy |
| EGRESS_BYPASS | 0x40 | 1 | Egress Bypass |
| EGRESS_ARB_WINS | 0x41 | 1 | Egress ARB Wins |
| EGRESS_ARB_LOSSES | 0x42 | 1 | Egress ARB Losses |
| EGRESS_STARVED | 0x43 | 1 | Egress Cycles in Starvation |
| RBOX_HOM_BYPASS | 0x50 | 1 | R-Box HOM Bypass |
| RBOX_SNP_BYPASS | 0x51 | 1 | R-Box SNP Bypass |
| S2B_HOM_BYPASS | 0x52 | 1 | S-Box to B-Box HOM Bypass |
| B2S_DRS_BYPASS | 0x53 | 1 | B-Box to S-Box DRS Bypass |
| BBOX_HOM_BYPASS | 0x54 | 1 | B-Box HOM Bypass |
| PKTS_SENT_HOM | 0x60 | 1 | HOM Packets Sent to System |
| PKTS_SENT_SNP | 0x62 | 1 | SNP Packets Sent to System |
| PKTS_SENT_NDR | 0x63 | 1 | NDR Packets Sent to System |
| PKTS_SENT_DRS | 0x64 | 1 | DRS Packets Sent to System |
| FLITS_SENT_DRS | 0x65 | 1 | DRS Flits Sent to System |
| PKTS_SENT_NCS | 0x66 | 1 | NCS Packets Sent to System |
| FLITS_SENT_NCS | 0x67 | 1 | NCS Flits Sent to System |
| PKTS_SENT_NCB | 0x68 | 1 | NCB Packets Sent to System |
| FLITS_SENT_NCB | 0x69 | 1 | NCB Flits Sent to System |

Table 2-37. Performance Monitor Events for S-Box Events

| Symbol Name | Event Code | Max Inc/ Cyc | Description |
|-------------------------------|------------|-----------------|---|
| RBOX_CREDIT_RETURNS | 0x6A | 1 | R-Box Credit Returns |
| BBOX_CREDIT_RETURNS | 0x6B | 1 | B-Box Credit Returns |
| TO_R_B_REQUESTS | 0x6C | 1 | System Bound Requests |
| PKTS_RCVD_NDR | 0x70 | 1 | NDR Packets Received from System |
| PKTS_RCVD_SNP | 0x71 | 1 | SNP Packets Received from System |
| PKTS_RCVD_DRS_FROM_R | 0x72 | 1 | DRS Packets Received from R-Box |
| PKTS_RCVD_DRS_FROM_B | 0x73 | 1 | DRS Packets Received from B-Box |
| PKTS_RCVD_NCS | 0x74 | 1 | NCS Packets Received from System |
| PKTS_RCVD_NCB | 0x75 | 1 | NCB Packets Received from System |
| RBOX_CREDITS | 0x76 | 1 | R-Box Credit Carrying Flits |
| BBOX_CREDITS | 0x77 | 1 | B-Box Credit Carrying Flits |
| NO_CREDIT_HOM | 0x80 | 1 | HOM Credit Unavailable |
| NO_CREDIT_SNP | 0x81 | 1 | SNP Credit Unavailable |
| NO_CREDIT_DRS | 0x82 | 1 | DRS Credit Unavailable |
| NO_CREDIT_NCS | 0x83 | 1 | NCS Credit Unavailable |
| NO_CREDIT_NCB | 0x84 | 1 | NCB Credit Unavailable |
| NO_CREDIT_NDR | 0x85 | 1 | NDR Credit Unavailable |
| NO_CREDIT_VNA | 0x86 | 1 | VNA Credit Unavailable |
| NO_CREDIT_AD | 0x87 | 1 | AD Credit Unavailable |
| NO_CREDIT_AK | 0x88 | 1 | AK Credit Unavailable |
| NO_CREDIT_BL | 0x89 | 1 | BL Credit Unavailable |
| NO_CREDIT_IPQ | 0x8A | 1 | IPQ Credit Unavailable |
| NO_CREDIT_LOC_NCS | 0x8B | 1 | Local NCS Credit Unavailable |
| TO_R_LOC_NCS_MSGQ_CYCLES_FULL | 0x8C | 1 | Cycles System Bound Local NCS Message Queue Full |
| TO_R_LOC_NCS_MSGQ_CYCLES_NE | 0x8D | 1 | Cycles System Bound Local NCS Message Queue Not Empty |
| TO_R_LOC_NCS_MSGQ_OCCUPANCY | 0x8E | 64 | System Bound Local NCS Message Queue Occupancy |
| PKTS_RCVD_LOC_NCS | 0x8F | 1 | Local NCS Packets Received from System |
| FLITS_SENT_LOC_NCS | 0x90 | 1 | Local NCS Flits Sent to System |

2.5.6 S-Box Performance Monitor Event List

This section enumerates Intel® Xeon® Processor E7 Family uncore performance monitoring events for the S-Box.

B2S_DRS_BYPASS

- **Title:** B-Box to S-Box DRS Bypass
- **Category:** Ring Bound Enhancement
- **Event Code:** 0x53, **Max. Inc/Cyc:** 1,
- **Definition:** Number of cycles the B-Box to S-Box DRS channel bypass optimization was utilized. Includes cycles used to transmit message flits and credit carrying idle credit flits.

BBOX_CREDITS

- **Title:** B-Box Credit Carrying Flits
- **Category:** Ring Bound Transmission
- **Event Code:** 0x77, **Max. Inc/Cyc:** 1,
- **Definition:** Number credit carrying idle flits received from the B-Box.

BBOX_CREDIT_RETURNS

- **Title:** B-Box Credit Returns
- **Category:** System Bound Transmission
- **Event Code:** 0x6B, **Max. Inc/Cyc:** 1,
- **Definition:** Number credit return idle flits sent to the B-Box.

BBOX_HOM_BYPASS

- **Title:** B-Box HOM Bypass
- **Category:** System Bound Enhancement
- **Event Code:** 0x54, **Max. Inc/Cyc:** 1,
- **Definition:** B-Box HOM Bypass optimization utilized.

EGRESS_ARB_LOSSES

- **Title:** Egress ARB Losses
- **Category:** Ring Bound Credits
- **Event Code:** 0x42, **Max. Inc/Cyc:** 1,
- **Definition:** Egress Arbitration Losses.
- **NOTE:** Enabling multiple subevents in this category will result in the counter being increased by the number of selected subevents that occur in a given cycle. Because only one of the even/odd FIFOs can arbitrate to send onto the ring in each cycle, the event for the even/odd FIFOs in each direction are exclusive. The bypass event for each direction is the sum of the bypass events of the even/odd FIFOs.

| Extension | umask [15:8] | Description |
|-----------|--------------|-----------------------------|
| --- | b000000 | (*nothing will be counted*) |
| AD_CW | b000001 | AD Clockwise |
| AD_CCW | b000010 | AD Counter-Clockwise |
| AD | b000011 | AD |
| AK_CW | b000100 | AK Clockwise |
| AK_CCW | b001000 | AK Counter-Clockwise |
| AK | b001100 | AK |
| BL_CW | b010000 | BL Clockwise |
| BL_CCW | b100000 | BL Counter-Clockwise |
| BL | b110000 | BL |

EGRESS_ARB_WINS

- **Title:** Egress ARB Wins
- **Category:** Ring Bound Transmission
- **Event Code:** 0x41, **Max. Inc/Cyc:** 1,
- **Definition:** Egress Arbitration Wins.
- **NOTE:** Enabling multiple subevents in this category will result in the counter being increased by the number of selected subevents that occur in a given cycle. Because only one of the even/odd FIFOs can arbitrate to send onto the ring in each cycle, the event for the even/odd FIFOs in each direction

are exclusive. The bypass event for each direction is the sum of the bypass events of the even/odd FIFOs.

| Extension | umask [15:8] | Description |
|-----------|--------------|-----------------------------|
| --- | b000000 | (*nothing will be counted*) |
| AD_CW | b000001 | AD Clockwise |
| AD_CCW | b000010 | AD Counter-Clockwise |
| AD | b000011 | AD |
| AK_CW | b000100 | AK Clockwise |
| AK_CCW | b001000 | AK Counter-Clockwise |
| AK | b001100 | AK |
| BL_CW | b010000 | BL Clockwise |
| BL_CCW | b100000 | BL Counter-Clockwise |
| BL | b110000 | BL |

EGRESS_BYPASS

- **Title:** Egress Bypass
- **Category:** Ring Bound Enhancement
- **Event Code:** 0x40, **Max. Inc/Cyc:** 1,
- **Definition:** Egress Bypass optimization utilized.
- **NOTE:** Enabling multiple subevents in this category will result in the counter being increased by the number of selected subevents that occur in a given cycle. Because only one of the even/odd FIFOs can arbitrate to send onto the ring in each cycle, the event for the even/odd FIFOs in each direction are exclusive. The bypass event for each direction is the sum of the bypass events of the even/odd FIFOs.

| Extension | umask [15:8] | Description |
|-----------|--------------|-----------------------------|
| --- | b000000 | (*nothing will be counted*) |
| AD_CW | b000001 | AD Clockwise |
| AD_CCW | b000010 | AD Counter-Clockwise |
| AD | b000011 | AD |
| AK_CW | b000100 | AK Clockwise |
| AK_CCW | b001000 | AK Counter-Clockwise |
| AK | b001100 | AK |
| BL_CW | b010000 | BL Clockwise |
| BL_CCW | b100000 | BL Counter-Clockwise |
| BL | b110000 | BL |

EGRESS_STARVED

- **Title:** Egress Cycles in Starvation
- **Category:** Ring Bound Credits
- **Event Code:** 0x43, **Max. Inc/Cyc:** 1,
- **Definition:** Number of cycles the S-Box egress FIFOs are in starvation.
- **NOTE:** Enabling multiple subevents in this category will result in the counter being increased by the number of selected subevents that occur in a given cycle. Because only one of the even/odd FIFOs can arbitrate to send onto the ring in each cycle, the event for the even/odd FIFOs in each direction

are exclusive. The bypass event for each direction is the sum of the bypass events of the even/odd FIFOs.

| Extension | umask [15:8] | Description |
|-----------|--------------|-----------------------------|
| --- | b000000 | (*nothing will be counted*) |
| AD_CW | b000001 | AD Clockwise |
| AD_CCW | b000010 | AD Counter-Clockwise |
| AD | b000011 | AD |
| AK_CW | b000100 | AK Clockwise |
| AK_CCW | b001000 | AK Counter-Clockwise |
| AK | b001100 | AK |
| BL_CW | b010000 | BL Clockwise |
| BL_CCW | b100000 | BL Counter-Clockwise |
| BL | b110000 | BL |

FLITS_SENT_DRS

- **Title:** DRS Flits Sent to System
- **Category:** System Bound Transmission
- **Event Code:** 0x65, **Max. Inc/Cyc:** 1,
- **Definition:** Number of data response flits the S-Box has transmitted to the system.

FLITS_SENT_LOC_NCS

- **Title:** Local NCS Flits Sent to System
- **Category:** System Bound Transmission
- **Event Code:** 0x90, **Max. Inc/Cyc:** 1,
- **Definition:** Number of local non-coherent standard flits the S-Box has transmitted to the system.

FLITS_SENT_NCB

- **Title:** NCB Flits Sent to System
- **Category:** System Bound Transmission
- **Event Code:** 0x69, **Max. Inc/Cyc:** 11,
- **Definition:** Number of non-coherent bypass flits the S-Box has transmitted to the system.

FLITS_SENT_NCS

- **Title:** NCS Flits Sent to System
- **Category:** System Bound Transmission
- **Event Code:** 0x67, **Max. Inc/Cyc:** 1,
- **Definition:** Number of non-coherent standard flits the S-Box has transmitted to the system.

HALFLINE_BYPASS

- **Title:** Half Cacheline Bypass
- **Category:** Ring Bound Enhancement
- **Event Code:** 0x30, **Max. Inc/Cyc:** 1,
- **Definition:** Half Cacheline Bypass optimization (where the line is sent early) was utilized.

NO_CREDIT_AD

- **Title:** AD Ring Credit Unavailable
- **Category:** Ring Bound Credits
- **Event Code:** 0x87, **Max. Inc/Cyc:** 1,
- **Definition:** Number of times the S-Box has a pending SNP, NCS or NCB message to send and there is no credit for the target egress FIFO.

NO_CREDIT_AK

- **Title:** AK Ring Credit Unavailable
- **Category:** Ring Bound Credits
- **Event Code:** 0x88, **Max. Inc/Cyc:** 1,
- **Definition:** Number of times the S-Box has a pending NDR or S2C credit return message to send but there is no credit for the target egress FIFO.

NO_CREDIT_BL

- **Title:** BL Ring Credit Unavailable
- **Category:** Ring Bound Credits
- **Event Code:** 0x89, **Max. Inc/Cyc:** 1,
- **Definition:** Number of times the S-Box has a pending DRS or debug message to send and there is no credit for the target egress FIFO.

NO_CREDIT_DRS

- **Title:** DRS Credit Unavailable
- **Category:** System Bound Credits
- **Event Code:** 0x82, **Max. Inc/Cyc:** 1,
- **Definition:** Number of times the S-Box has a pending data response message to send and there is no DRS or VNA credit available.

NO_CREDIT_HOM

- **Title:** HOM Credit Unavailable
- **Category:** System Bound Credits
- **Event Code:** 0x80, **Max. Inc/Cyc:** 1,
- **Definition:** Number of times the S-Box has a pending home message to send and there is no HOM or VNA credit available.

NO_CREDIT_IPQ

- **Title:** IPQ Credit Unavailable
- **Category:** Ring Bound Credits
- **Event Code:** 0x8A, **Max. Inc/Cyc:** 1,
- **Definition:** Number of times the S-Box has an incoming SNP to send but there is no IPQ credit available for the target C-Box.

NO_CREDIT_LOC_NCS

- **Title:** Local NCS Credit Unavailable
- **Category:** System Bound Credits
- **Event Code:** 0x8B, **Max. Inc/Cyc:** 1,
- **Definition:** Number of times the S-Box has a pending local non-coherent standard message to send and there is no NCS or VNA credit available.

NO_CREDIT_NCB

- **Title:** NCB Credit Unavailable
- **Category:** System Bound Credits
- **Event Code:** 0x84, **Max. Inc/Cyc:** 1,
- **Definition:** Number of times the S-Box has a pending non-coherent bypass message to send and there is no NCB or VNA credit available.

NO_CREDIT_NCS

- **Title:** NCS Credit Unavailable
- **Category:** System Bound Credits
- **Event Code:** 0x83, **Max. Inc/Cyc:** 1,
- **Definition:** Number of times the S-Box has a pending non-coherent standard message to send and there is no NCS or VNA credit available.

NO_CREDIT_NDR

- **Title:** NDR Credit Unavailable
- **Category:** System Bound Credits
- **Event Code:** 0x85, **Max. Inc/Cyc:** 1,
- **Definition:** Number of times the S-Box has a pending non-data response message to send and there is no NDR or VNA credit available.

NO_CREDIT_SNP

- **Title:** SNP Credit Unavailable
- **Category:** System Bound Credits
- **Event Code:** 0x81, **Max. Inc/Cyc:** 1,
- **Definition:** Number of times the S-Box has a pending snoop message to send and there is no SNP or VNA credit available.

NO_CREDIT_VNA

- **Title:** VNA Credit Unavailable
- **Category:** System Bound Transmission
- **Event Code:** 0x86, **Max. Inc/Cyc:** 1,
- **Definition:** Number of times the S-Box has exhausted its VNA credit pool. When more than one subevent is selected, the credit counter will be incremented by the number of selected subevents that occur in each cycle.

| Extension | umask [15:8] | Description |
|-----------|--------------|-----------------------------|
| --- | b00 | (*nothing will be counted*) |
| RBOX | b01 | R-Box Out |
| BBOX | b10 | B-Box Out |
| ALL | b11 | Both R and B Boxes |

PKTS_RCVD_DRS_FROM_B

- **Title:** DRS Packets Received from B-Box
- **Category:** Ring Bound Transmission
- **Event Code:** 0x73, **Max. Inc/Cyc:** 1,
- **Definition:** Number of data response packets the S-Box has received from the B-Box.
- **NOTE:** DRS messages are always full cacheline messages which are 9 flits. Multiply this event by 9 to derive flit traffic from the B-Box due to DRS messages.

PKTS_RCVD_DRS_FROM_R

- **Title:** DRS Packets Received from R-Box
- **Category:** Ring Bound Transmission
- **Event Code:** 0x72, **Max. Inc/Cyc:** 9,
- **Definition:** Number of data response packets the S-Box has received from the R-Box.
- **NOTE:** DRS messages are always full cacheline messages which are 9 flits. Multiply this event by 9 to derive flit traffic from the R-Box due to DRS messages.

PKTS_RCVD_LOC_NCS

- **Title:** Local NCS Packets Received from System
- **Category:** Ring Bound Transmission
- **Event Code:** 0x8F, **Max. Inc/Cyc:** 1,
- **Definition:** Number of local non-coherent standard packets the S-Box has received from the system.
- **NOTE:** The only ring bound local NCS message type is NcMsgS (StopReq1). There are always 3 flits. Multiply this event by 3 to derive flit traffic from the system due to local NCS messages.

PKTS_RCVD_NCB

- **Title:** NCB Packets Received from System
- **Category:** Ring Bound Transmission
- **Event Code:** 0x75, **Max. Inc/Cyc:** 1,
- **Definition:** Number of non-coherent bypass packets the S-Box has received from the system.
- **NOTE:** The only ring bound NCB message types are: NcMsgB (StartReq2, VLW), IntLogical, IntPhysical. These are all 11 flit messages. Multiply this event by 11 to derive flit traffic from the system due to NCB messages.

PKTS_RCVD_NCS

- **Title:** NCS Packets Received from System
- **Category:** Ring Bound Transmission
- **Event Code:** 0x74, **Max. Inc/Cyc:** 1,
- **Definition:** Number of non-coherent standard packets the S-Box has received from the system.
- **NOTE:** The only ring bound NCS message type is NcMsgS (StopReq1). There are always 3 flits. Multiply this event by 3 to derive flit traffic from the system due to NCS messages.

PKTS_RCVD_NDR

- **Title:** NDR Packets Received from System
- **Category:** Ring Bound Transmission
- **Event Code:** 0x70, **Max. Inc/Cyc:** 1,
- **Definition:** Number of non-data response packets the S-Box has received from the system.

PKTS_RCVD_SNP

- **Title:** SNP Packets Received from System
- **Category:** Ring Bound Transmission
- **Event Code:** 0x71, **Max. Inc/Cyc:** 1,
- **Definition:** Number of snoop packets the S-Box has received from the system.

PKTS_SENT_DRS

- **Title:** DRS Packets Sent to System
- **Category:** System Bound Transmission
- **Event Code:** 0x64, **Max. Inc/Cyc:** 1,
- **Definition:** Number of DRS packets the S-Box has transmitted to the system.
- **NOTE:** If multiple C-Boxes are selected, this event counts the total data response packets sent by all the selected C-Boxes. In the cases where one DRS message spawns two messages, one to the

requester and one to the home, this event only counts the first DRS message. DRS messages are always full cacheline messages which are 9 flits.

| Extension | umask [15:8] | Description |
|-----------|--------------|-----------------------------|
| --- | b00000 | (*nothing will be counted*) |
| CBOX0_5 | bxxxx1 | CBOX 0 and 5 |
| CBOX1_6 | bxxx1x | CBOX 1 and 6 |
| CBOX2_7 | bxx1xx | CBOX 2 and 7 |
| CBOX3_8 | bx1xxx | CBOX 3 and 8 |
| CBOX4_9 | b1xxxx | CBOX 4 and 9 |
| ALL | b11111 | All C-Boxes |

PKTS_SENT_HOM

- **Title:** HOM Packets Sent to System
- **Category:** System Bound Transmission
- **Event Code:** 0x60, **Max. Inc/Cyc:** 1,
- **Definition:** Number of home packets the S-Box has transmitted to the R-Box or B-Box. If both R-Box and B-Box are selected, counts the total number of home packets sent to both boxes.

| Extension | umask [15:8] | Description |
|-----------|--------------|-----------------------------|
| --- | b00 | (*nothing will be counted*) |
| RBOX | b01 | R-Box (1 flit per request) |
| BBOX | b10 | B-Box (1 flit per request) |
| ALL | b11 | Both R and B Boxes |

PKTS_SENT_NCB

- **Title:** NCB Packets Sent to System
- **Category:** System Bound Transmission
- **Event Code:** 0x68, **Max. Inc/Cyc:** 11,
- **Definition:** Number of NCB packets the S-Box has transmitted to the system.
- **NOTE:** If multiple C-Boxes are selected, this event counts the total non-coherent bypass packets sent by all the selected C-Boxes. The only ring bound NCB message types are: NcMsgB (StartReq2, VLW), IntLogical, IntPhysical. These are all 11 flit messages.

| Extension | umask [15:8] | Description |
|-----------|--------------|-----------------------------|
| --- | b00000 | (*nothing will be counted*) |
| CBOX0_5 | bxxxx1 | CBOX 0 and 5 |
| CBOX1_6 | bxxx1x | CBOX 1 and 6 |
| CBOX2_7 | bxx1xx | CBOX 2 and 7 |
| CBOX3_8 | bx1xxx | CBOX 3 and 8 |
| CBOX4_9 | b1xxxx | CBOX 4 and 9 |
| ALL | b11111 | All C-Boxes |

PKTS_SENT_NCS

- **Title:** NCS Packets Sent to System
- **Category:** System Bound Transmission
- **Event Code:** 0x66, **Max. Inc/Cyc:** 3,
- **Definition:** Number of NCS packets the S-Box has transmitted to the system.
- **NOTE:** If multiple C-Boxes are selected, this event counts the total non-coherent standard packets sent by all the selected C-Boxes. The only ring bound NCS message type is NcMsgS (StopReq1). There are always 3 flits.

| Extension | umask [15:8] | Description |
|-----------|--------------|-----------------------------|
| --- | b00000 | (*nothing will be counted*) |
| CBOX0_5 | bxxxx1 | CBOX 0 and 5 |
| CBOX1_6 | bxxx1x | CBOX 1 and 6 |
| CBOX2_7 | bxx1xx | CBOX 2 and 7 |
| CBOX3_8 | bx1xxx | CBOX 3 and 8 |
| CBOX4_9 | b1xxxx | CBOX 4 and 9 |
| ALL | b11111 | All C-Boxes |

PKTS_SENT_NDR

- **Title:** NDR Packets Sent to System
- **Category:** System Bound Transmission
- **Event Code:** 0x63, **Max. Inc/Cyc:** 1,
- **Definition:** Number of non-data response packets the S-Box has transmitted to the system.

PKTS_SENT_SNP

- **Title:** SNP Packets Sent to System
- **Category:** System Bound Transmission
- **Event Code:** 0x62, **Max. Inc/Cyc:** 1,
- **Definition:** Number of SNP packets the S-Box has transmitted to the system. This event only counts the first snoop that is spawned from a home request. When S-Box broadcast is enabled, this event does not count the additional snoop packets that are spawned.

RBOX_CREDIT_CARRIERS

- **Title:** R-Box Credit Carrying Flits
- **Category:** Ring Bound Transmission
- **Event Code:** 0x76, **Max. Inc/Cyc:** 1,
- **Definition:** Number credit carrying idle flits received from the R-Box.

RBOX_CREDIT_RETURNS

- **Title:** R-Box Credit Returns
- **Category:** System Bound Transmission
- **Event Code:** 0x6A, **Max. Inc/Cyc:** 1,
- **Definition:** Number credit return idle flits sent to the R-Box.

RBOX_HOM_BYPASS

- **Title:** R-Box HOM Bypass
- **Category:** System Bound Enhancement
- **Event Code:** 0x50, **Max. Inc/Cyc:** 1,
- **Definition:** R-Box HOM Bypass optimization was utilized.

RBOX_SNP_BYPASS

- **Title:** R-Box SNP Bypass
- **Category:** System Bound Enhancement
- **Event Code:** 0x51, **Max. Inc/Cyc:** 1,
- **Definition:** R-Box SNP bypass optimization utilized. When both snoop and big snoop bypass are selected, the performance counter will increment on both subevents.

| Extension | umask [15:8] | Description |
|-----------|--------------|-----------------------------|
| --- | b00 | (*nothing will be counted*) |
| SNP | b01 | Snoop |
| BIG_SNP | b10 | Big Snoop |
| ALL | b11 | Both Bypasses |

REQ_TBL_OCCUPANCY

- **Title:** Request Table Occupancy
- **Category:** Ring Bound Queue
- **Event Code:** 0x31, **Max. Inc/Cyc:** 48,
- **Definition:** Number of request table entries occupied by socket requests. Local means request is targeted towards the B-Boxes in the same socket. Requests to the B-Box in the same socket are considered remote.
- **NOTE:** Occupancy is tracked from allocation to deallocation of each entry in the queue.

| Extension | umask [15:8] | Description |
|-----------|--------------|-----------------------------|
| --- | b00 | (*nothing will be counted*) |
| LOCAL | b01 | Local |
| REMOTE | b10 | Remote |
| ALL | b11 | Local And Remote |

S2B_HOM_BYPASS

- **Title:** S-Box to B-Box HOM Bypass
- **Category:** System Bound Enhancement
- **Event Code:** 0x52, **Max. Inc/Cyc:** 1,
- **Definition:** Number of cycles the S-Box to B-Box HOM channel bypass optimization was utilized. Includes cycles used to transmit message flits and credit carrying idle credit flits.

TO_RING_B2S_MSGQ_CYCLES_FULL

- **Title:** Ring Bound B2S Message Queue Full
- **Category:** Ring Bound Queue
- **Event Code:** 0x2B, **Max. Inc/Cyc:** 1,
- **Definition:** Number of cycles in which the header buffer, containing B to S-Box messages on their way to the Ring, is full.

TO_RING_B2S_MSGQ_CYCLES_NE

- **Title:** Cycles Ring Bound B2S Message Queue Not Empty
- **Category:** Ring Bound Queue
- **Event Code:** 0x2D, **Max. Inc/Cyc:** 1,
- **Definition:** Number of cycles in which the header buffer, containing B to S-Box messages on their way to the Ring, has one or more entries allocated.

TO_RING_B2S_MSGQ_OCCUPANCY

- **Title:** Ring Bound B2S Message Queue Occupancy
- **Category:** Ring Bound Queue
- **Event Code:** 0x2F, **Max. Inc/Cyc:** 8,
- **Definition:** Number of entries in header buffer containing B to S-Box messages on their way to the Ring.

TO_RING_MSGQ_OCCUPANCY

- **Title:** Ring Bound Message Queue Occupancy
- **Category:** Ring Bound Queue
- **Event Code:** 0x26, **Max. Inc/Cyc:** 1,
- **Definition:** Number of entries in header buffer containing SNP, NCS or NCB messages headed for the Ring. Each subevent represents usage of the buffer by a particular message class. When more than one message class is selected, the queue occupancy counter counts the total number of buffer entries occupied by messages in the selected message classes.
- **NOTE:** Total of the buffer entries occupied by all message classes in umask will never exceed 36.

| Extension | umask [15:8] | Description |
|-----------|--------------|--|
| --- | b000 | (*nothing will be counted*) |
| SNP | bxx1 | Snoop (31 entry buffer) |
| NCS | bx1x | Non-coherent Standard (4 entry buffer) |
| NCB | b1xx | Non-Coherent Bypass (4 entry buffer) |
| ALL | b111 | All C-Boxes |

TO_RING_NCB_MSGQ_CYCLES_FULL

- **Title:** Cycles Ring Bound NCB Message Queue Full
- **Category:** Ring Bound Queue
- **Event Code:** 0x21, **Max. Inc/Cyc:** 1,
- **Definition:** Number of cycles in which the header buffer, containing NCB messages on their way to the Ring, is full.

TO_RING_NCB_MSGQ_CYCLES_NE

- **Title:** Cycles Ring Bound NCB Message Queue Not Empty
- **Category:** Ring Bound Queue
- **Event Code:** 0x24, **Max. Inc/Cyc:** 1,
- **Definition:** Number of cycles in which the header buffer, containing NCB messages on their way to the Ring, has one or more entries allocated.

TO_RING_NCS_MSGQ_CYCLES_FULL

- **Title:** Cycles Ring Bound NCS Message Queue Full
- **Category:** Ring Bound Queue
- **Event Code:** 0x22, **Max. Inc/Cyc:** 1,
- **Definition:** Number of cycles in which the header buffer, containing NCS messages on their way to the Ring, is full.

TO_RING_NDR_MSGQ_CYCLES_FULL

- **Title:** Cycles Ring Bound NDR Message Queue Full
- **Category:** Ring Bound Queue
- **Event Code:** 0x27, **Max. Inc/Cyc:** 1,
- **Definition:** Number of cycles in which the header buffer, containing NDR messages on their way to the Ring, is full.

TO_RING_NDR_MSGQ_CYCLES_NE

- **Title:** Cycles Ring Bound NDR Message Queue Not Empty
- **Category:** Ring Bound Queue
- **Event Code:** 0x28, **Max. Inc/Cyc:** 1,
- **Definition:** Number of cycles in which the header buffer, containing NDR messages on their way to the Ring, has one or more entries allocated.

TO_RING_NDR_MSGQ_OCCUPANCY

- **Title:** Ring Bound SNP Message Queue Occupancy
- **Category:** Ring Bound Queue
- **Event Code:** 0x29, **Max. Inc/Cyc:** 32,
- **Definition:** Number of entries in header buffer containing NDR messages on their way to the Ring.

TO_RING_R2S_MSGQ_CYCLES_FULL

- **Title:** Cycles Ring Bound R2S Message Queue Full
- **Category:** Ring Bound Queue
- **Event Code:** 0x2A, **Max. Inc/Cyc:** 1,
- **Definition:** Number of cycles in which the header buffer, containing R to S-Box messages on their way to the Ring, is full.

TO_RING_R2S_MSGQ_CYCLES_NE

- **Title:** Cycles Ring Bound R2S Message Queue Not Empty
- **Category:** Ring Bound Queue
- **Event Code:** 0x2C, **Max. Inc/Cyc:** 1,
- **Definition:** Number of cycles in which the header buffer, containing R to S-Box messages on their way to the Ring, has one or more entries allocated.

TO_RING_R2S_MSGQ_OCCUPANCY

- **Title:** Ring Bound R2S Message Queue Occupancy
- **Category:** Ring Bound Queue
- **Event Code:** 0x2E, **Max. Inc/Cyc:** 8,
- **Definition:** Number of entries in header buffer containing R to S messages on their way to the Ring.

TO_RING_SNP_MSGQ_CYCLES_FULL

- **Title:** Cycles Ring Bound SNP Message Queue Full
- **Category:** Ring Bound Queue
- **Event Code:** 0x20, **Max. Inc/Cyc:** 1,
- **Definition:** Number of cycles in which the header buffer, containing SNP messages on their way to the Ring, is full.

TO_RING_SNP_MSGQ_CYCLES_NE

- **Title:** Cycles Ring Bound SNP Message Queue Not Empty
- **Category:** Ring Bound Queue
- **Event Code:** 0x23, **Max. Inc/Cyc:** 1,
- **Definition:** Number of cycles in which the header buffer, containing SNP messages on their way to the Ring, has one or more entries allocated.

TO_R_DRS_MSGQ_CYCLES_FULL

- **Title:** Cycles System Bound DRS Message Queue Full.
- **Category:** System Bound Queue
- **Event Code:** 0x0E, **Max. Inc/Cyc:** 1,
- **Definition:** Number of cycles in which the header buffer for the selected C-Box, containing DRS messages heading to a System Agent (through the R-Box), is full. Only one C-Box's DRS header buffer should be selected for the buffer full checking to be correct, else the result is undefined.

| Extension | umask [15:8] | Description |
|-----------|--------------|-----------------------------|
| --- | b00000 | (*nothing will be counted*) |
| CBOX0_5 | bxxxx1 | CBOX 0 and 5 |
| CBOX1_6 | bxxx1x | CBOX 1 and 6 |
| CBOX2_7 | bxx1xx | CBOX 2 and 7 |
| CBOX3_8 | bx1xxx | CBOX 3 and 8 |
| CBOX4_9 | b1xxxx | CBOX 4 and 9 |
| ALL | b11111 | All C-Boxes |

TO_R_DRS_MSGQ_CYCLES_NE

- **Title:** Cycles System Bound DRS Message Queue Not Empty
- **Category:** System Bound Queue
- **Event Code:** 0x0F, **Max. Inc/Cyc:** 1,
- **Definition:** Number of cycles in which the header buffer for the selected C-Box, containing DRS messages heading to a System Agent (through the R-Box), has one or more entries allocated. When more than one C-Box is selected, the event is asserted when any of the selected C-Box DRS header buffers are not empty.

| Extension | umask [15:8] | Description |
|-----------|--------------|-----------------------------|
| --- | b00000 | (*nothing will be counted*) |
| CBOX0_5 | bxxxx1 | CBOX 0 and 5 |
| CBOX1_6 | bxxx1x | CBOX 1 and 6 |
| CBOX2_7 | bxx1xx | CBOX 2 and 7 |
| CBOX3_8 | bx1xxx | CBOX 3 and 8 |
| CBOX4_9 | b1xxxx | CBOX 4 and 9 |
| ALL | b11111 | All C-Boxes |

TO_R_DRS_MSGQ_OCCUPANCY

- **Title:** System Bound DRS Message Queue Occupancy
- **Category:** System Bound Queue
- **Event Code:** 0x10, **Max. Inc/Cyc:** 16,
- **Definition:** Number of entries in the header buffer for the selected C-Box, containing DRS messages heading to a System Agent (through the R-Box). When more than one C-Box is selected, the queue occupancy counter counts the total number of occupied entries in all selected C-Box DRS header buffers.
- **NOTE:** 1 buffer per C-Box, 4 entries each.

| Extension | umask [15:8] | Description |
|-----------|--------------|-----------------------------|
| --- | b00000 | (*nothing will be counted*) |
| CBOX0_5 | bxxxx1 | CBOX 0 and 5 |
| CBOX1_6 | bxxx1x | CBOX 1 and 6 |
| CBOX2_7 | bxx1xx | CBOX 2 and 7 |
| CBOX3_8 | bx1xxx | CBOX 3 and 8 |
| CBOX4_9 | b1xxxx | CBOX 4 and 9 |
| ALL | b11111 | All C-Boxes |

TO_R_B_HOM_MSGQ_CYCLES_FULL

- **Title:** Cycles System Bound HOM Message Queue Full.
- **Category:** System Bound Queue
- **Event Code:** 0x03, **Max. Inc/Cyc:** 1,
- **Definition:** Number of cycles in which the header buffer, containing HOM messages heading to a System Agent (through the B or R-Box), is full. If both R-Box and B-Box subevents are selected, this event is asserted when the total number of entries in the R-Box and B-Box Home header buffers is equal to 64.

| Extension | umask [15:8] | Description |
|-----------|--------------|-----------------------------|
| --- | b00 | (*nothing will be counted*) |
| RBOX | b01 | R-Box |
| BBOX | b10 | B-Box |
| RBBOX | b11 | R or B-Box |

TO_R_B_HOM_MSGQ_CYCLES_NE

- **Title:** Cycles System Bound HOM Header Not Empty
- **Category:** System Bound Queue
- **Event Code:** 0x06, **Max. Inc/Cyc:** 1,
- **Definition:** Number of cycles in which the header buffer, containing HOM messages heading to a System Agent (through the B or R-Box), has one or more entries allocated. If both R-Box and B-Box subevents are selected, this event is asserted when the total number of entries in the R-Box and B-Box Home header buffers is equal to 64.

| Extension | umask [15:8] | Description |
|-----------|--------------|-----------------------------|
| --- | b00 | (*nothing will be counted*) |
| RBOX | b01 | R-Box |
| BBOX | b10 | B-Box |
| RBBOX | b11 | R or B-Box |

TO_R_B_HOM_MSGQ_OCCUPANCY

- **Title:** System Bound HOM Message Queue Occupancy
- **Category:** System Bound Queue
- **Event Code:** 0x07, **Max. Inc/Cyc:** 64,
- **Definition:** Number of entries in the header buffer containing HOM messages heading to a System Agent (through the B or R-Box).
- **NOTE:** 1 buffer for the R-Box and 1 for the B-Box, 64 entries each. The sum of the occupied entries in the 2 header buffers will never exceed 64.

| Extension | umask [15:8] | Description |
|-----------|--------------|-----------------------------|
| --- | b00 | (*nothing will be counted*) |
| RBOX | b01 | R-Box |
| BBOX | b10 | B-Box |
| RBBOX | b11 | R or B-Box |

TO_R_LOC_NCS_MSGQ_CYCLES_FULL

- **Title:** Cycles System Bound Local NCS Message Queue Full.
- **Category:** System Bound Queue
- **Event Code:** 0x8C, **Max. Inc/Cyc:** 1,
- **Definition:** Number of cycles in which the header buffer for the selected C-Box, containing Local NCS messages heading to a System Agent (through the R-Box), is full. Only one C-Box's Local NCS header buffer should be selected for the buffer full checking to be correct, else the result is undefined.

| Extension | umask [15:8] | Description |
|-----------|--------------|-----------------------------|
| --- | b00000 | (*nothing will be counted*) |
| CBOX0_5 | bxxxx1 | CBOX 0 and 5 |
| CBOX1_6 | bxxx1x | CBOX 1 and 6 |
| CBOX2_7 | bxx1xx | CBOX 2 and 7 |
| CBOX3_8 | bx1xxx | CBOX 3 and 8 |
| CBOX4_9 | b1xxxx | CBOX 4 and 9 |
| ALL | b11111 | All C-Boxes |

TO_R_LOC_NCS_MSGQ_CYCLES_NE

- **Title:** Cycles System Bound Local NCS Message Queue Not Empty
- **Category:** System Bound Queue
- **Event Code:** 0x8D, **Max. Inc/Cyc:** 1,
- **Definition:** Number of cycles in which the header buffer for the selected C-Box, containing Local NCS messages heading to a System Agent (through the R-Box), has one or more entries allocated. When more than one C-Box is selected, the event is asserted when any of the selected C-Box Local NCS header buffers are not empty.

| Extension | umask [15:8] | Description |
|-----------|--------------|-----------------------------|
| --- | b00000 | (*nothing will be counted*) |
| CBOX0_5 | bxxxx1 | CBOX 0 and 5 |
| CBOX1_6 | bxxx1x | CBOX 1 and 6 |
| CBOX2_7 | bxx1xx | CBOX 2 and 7 |
| CBOX3_8 | bx1xxx | CBOX 3 and 8 |
| CBOX4_9 | b1xxxx | CBOX 4 and 9 |
| ALL | b11111 | All C-Boxes |

TO_R_LOC_NCS_MSGQ_OCCUPANCY

- **Title:** System Bound Local NCS Message Queue Occupancy
- **Category:** System Bound Queue
- **Event Code:** 0x8E, **Max. Inc/Cyc:** 2,
- **Definition:** Number of entries in the header buffer for the selected C-Box, containing Local NCS messages heading to a System Agent (through the R-Box). When more than one C-Box is selected, the queue occupancy counter counts the total number of occupied entries in all selected C-Box Local NCS header buffers.
- **NOTE:** 1 buffer per C-Box, 2 entries each.

| Extension | umask [15:8] | Description |
|-----------|--------------|-----------------------------|
| --- | b00000 | (*nothing will be counted*) |
| CBOX0_5 | bxxxx1 | CBOX 0 and 5 |
| CBOX1_6 | bxxx1x | CBOX 1 and 6 |
| CBOX2_7 | bxx1xx | CBOX 2 and 7 |
| CBOX3_8 | bx1xxx | CBOX 3 and 8 |
| CBOX4_9 | b1xxxx | CBOX 4 and 9 |
| ALL | b11111 | All C-Boxes |

TO_R_NCB_MSGQ_CYCLES_FULL

- **Title:** Cycles System Bound NCB Message Queue Full.
- **Category:** System Bound Queue
- **Event Code:** 0x11, **Max. Inc/Cyc:** 1,
- **Definition:** Number of cycles in which the header buffer for the selected C-Box, containing NCB messages heading to a System Agent (through the R-Box), is full. Only one C-Box's NCB header buffer should be selected for the buffer full checking to be correct, else the result is undefined.

| Extension | umask [15:8] | Description |
|-----------|--------------|-----------------------------|
| --- | b00000 | (*nothing will be counted*) |
| CBOX0_5 | bxxxx1 | CBOX 0 and 5 |
| CBOX1_6 | bxxx1x | CBOX 1 and 6 |
| CBOX2_7 | bxx1xx | CBOX 2 and 7 |
| CBOX3_8 | bx1xxx | CBOX 3 and 8 |
| CBOX4_9 | b1xxxx | CBOX 4 and 9 |
| ALL | b11111 | All C-Boxes |

TO_R_NCB_MSGQ_CYCLES_NE

- **Title:** Cycles System Bound NCB Message Queue Not Empty
- **Category:** System Bound Queue
- **Event Code:** 0x12, **Max. Inc/Cyc:** 1,
- **Definition:** Number of cycles in which the header buffer for the selected C-Box, containing NCB messages heading to a System Agent (through the R-Box), has one or more entries allocated. When more than one C-Box is selected, the event is asserted when any of the selected C-Box DRS header buffers are not empty.

| Extension | umask [15:8] | Description |
|-----------|--------------|-----------------------------|
| --- | b00000 | (*nothing will be counted*) |
| CBOX0_5 | bxxxx1 | CBOX 0 and 5 |
| CBOX1_6 | bxxx1x | CBOX 1 and 6 |
| CBOX2_7 | bxx1xx | CBOX 2 and 7 |
| CBOX3_8 | bx1xxx | CBOX 3 and 8 |
| CBOX4_9 | b1xxxx | CBOX 4 and 9 |
| ALL | b11111 | All C-Boxes |

TO_R_NCB_MSGQ_OCCUPANCY

- **Title:** System Bound NCB Message Queue Occupancy
- **Category:** System Bound Queue
- **Event Code:** 0x13, **Max. Inc/Cyc:** 8,
- **Definition:** Number of entries in the header buffer for the selected C-Box, containing NCB messages heading to a System Agent (through the R-Box). When more than one C-Box is selected, the queue occupancy counter counts the total number of occupied entries in all selected C-Box NCB header buffers.
- **NOTE:** 1 buffer per C-Box, 2 entries each.

| Extension | umask [15:8] | Description |
|-----------|--------------|-----------------------------|
| --- | b00000 | (*nothing will be counted*) |
| CBOX0_5 | bxxxx1 | CBOX 0 and 5 |
| CBOX1_6 | bxxx1x | CBOX 1 and 6 |
| CBOX2_7 | bxx1xx | CBOX 2 and 7 |
| CBOX3_8 | bx1xxx | CBOX 3 and 8 |
| CBOX4_9 | b1xxxx | CBOX 4 and 9 |
| ALL | b11111 | All C-Boxes |

TO_R_NCS_MSGQ_CYCLES_FULL

- **Title:** Cycles System Bound NCS Message Queue Full.
- **Category:** System Bound Queue
- **Event Code:** 0x14, **Max. Inc/Cyc:** 1,
- **Definition:** Number of cycles in which the header buffer for the selected C-Box, containing NCS messages heading to a System Agent (through the R-Box), is full. Only one C-Box's NCS header buffer should be selected for the buffer full checking to be correct, else the result is undefined.

| Extension | umask [15:8] | Description |
|-----------|--------------|-----------------------------|
| --- | b00000 | (*nothing will be counted*) |
| CBOX0_5 | bxxxx1 | CBOX 0 and 5 |
| CBOX1_6 | bxxx1x | CBOX 1 and 6 |
| CBOX2_7 | bxx1xx | CBOX 2 and 7 |
| CBOX3_8 | bx1xxx | CBOX 3 and 8 |
| CBOX4_9 | b1xxxx | CBOX 4 and 9 |
| ALL | b11111 | All C-Boxes |

TO_R_NCS_MSGQ_CYCLES_NE

- **Title:** Cycles System Bound NCS Message Queue Not Empty
- **Category:** System Bound Queue
- **Event Code:** 0x15, **Max. Inc/Cyc:** 1,
- **Definition:** Number of cycles in which the header buffer for the selected C-Box, containing NCS messages heading to a System Agent (through the R-Box), has one or more entries allocated. When more than one C-Box is selected, the event is asserted when any of the selected C-Box NCS header buffers are not empty.

| Extension | umask [15:8] | Description |
|-----------|--------------|-----------------------------|
| --- | b00000 | (*nothing will be counted*) |
| CBOX0_5 | bxxxx1 | CBOX 0 and 5 |
| CBOX1_6 | bxxx1x | CBOX 1 and 6 |
| CBOX2_7 | bxx1xx | CBOX 2 and 7 |
| CBOX3_8 | bx1xxx | CBOX 3 and 8 |
| CBOX4_9 | b1xxxx | CBOX 4 and 9 |
| ALL | b11111 | All C-Boxes |

TO_R_NCS_MSGQ_OCCUPANCY

- **Title:** System Bound NCS Message Queue Occupancy
- **Category:** System Bound Queue
- **Event Code:** 0x16, **Max. Inc/Cyc:** 2,
- **Definition:** Number of entries in the header buffer for the selected C-Box, containing NCS messages heading to a System Agent (through the R-Box). When more than one C-Box is selected, the queue occupancy counter counts the total number of occupied entries in all selected C-Box NCS header buffers.
- **NOTE:** 1 buffer per C-Box, 2 entries each.

| Extension | umask [15:8] | Description |
|-----------|--------------|-----------------------------|
| --- | b00000 | (*nothing will be counted*) |
| CBOX0_5 | bxxxx1 | CBOX 0 and 5 |
| CBOX1_6 | bxxx1x | CBOX 1 and 6 |
| CBOX2_7 | bxx1xx | CBOX 2 and 7 |
| CBOX3_8 | bx1xxx | CBOX 3 and 8 |
| CBOX4_9 | b1xxxx | CBOX 4 and 9 |
| ALL | b11111 | All C-Boxes |

TO_R_NDR_MSGQ_CYCLES_FULL

- **Title:** Cycles System Bound NDR Message Queue Full
- **Category:** System Bound Queue
- **Event Code:** 0x0B, **Max. Inc/Cyc:** 1,
- **Definition:** Number of cycles in which the header buffer, containing NDR messages heading to a System Agent (through the R-Box), is full.

TO_R_NDR_MSGQ_CYCLES_NE

- **Title:** Cycles System Bound NDR Message Queue Not Empty
- **Category:** System Bound Queue
- **Event Code:** 0x0C, **Max. Inc/Cyc:** 1,
- **Definition:** Number of cycles in which the header buffer, containing NDR messages heading to a System Agent (through the R-Box), has one or more entries allocated.

TO_R_NDR_MSGQ_OCCUPANCY

- **Title:** System Bound NDR Message Queue Occupancy
- **Category:** System Bound Queue
- **Event Code:** 0x0D, **Max. Inc/Cyc:** 16,
- **Definition:** Number of entries in the header buffer, containing NDR messages heading to a System Agent (through the R-Box).

TO_R_PROG_EV

- **Title:** System Bound Programmable Event
- **Category:** System Bound Queue
- **Event Code:** 0x00, **Max. Inc/Cyc:** 1,
- **Definition:** Programmable Event heading to a System Agent (through the R-Box). Match/Mask on criteria set in S_MSR_MATCH/MASK registers (Refer to [Section 2.5.3.4, "S-Box Registers for Mask/Match Facility"](#)).

TO_R_B_REQUESTS

- **Title:** System Bound Requests
- **Category:** System Bound Transmission
- **Event Code:** 0x6C, **Max. Inc/Cyc:** 1,
- **Definition:** Socket request (both B-Boxes). Local means request is targeted towards the B-Boxes in the same socket. Requests to the U-Box in the same socket are considered remote. The S-Box checks for Home NID[4:2] equal to S-Box NID[4:2] and Home NID[0] equal to 1.

| Extension | umask [15:8] | Description |
|-----------|--------------|-----------------------------|
| --- | b00 | (*nothing will be counted*) |
| LOCAL | b01 | Local |
| REMOTE | b10 | Remote |
| ALL | b11 | Local And Remote |

TO_R_SNP_MSGQ_CYCLES_FULL

- **Title:** Cycles System Bound SNP Message Queue Full
- **Category:** System Bound Queue
- **Event Code:** 0x08, **Max. Inc/Cyc:** 1,
- **Definition:** Number of cycles in which the header buffer, containing SNP messages heading to a System Agent (through the R-Box), is full.

TO_R_SNP_MSGQ_CYCLES_NE

- **Title:** Cycles System Bound SNP Message Queue Not Empty
- **Category:** System Bound Queue
- **Event Code:** 0x09, **Max. Inc/Cyc:** 1,
- **Definition:** Number of cycles in which the header buffer, containing SNP messages heading to a System Agent (through the R-Box), has one or more entries allocated.

TO_R_SNP_MSGQ_OCCUPANCY

- **Title:** System Bound SNP Message Queue Occupancy
- **Category:** System Bound Queue
- **Event Code:** 0x0A, **Max. Inc/Cyc:** 32,
- **Definition:** Number of entries in the header buffer, containing SNP messages heading to a System Agent (through the R-Box).

2.6 R-Box Performance Monitoring

2.6.1 Overview of the R-Box

The Crossbar Router (R-Box) is a 8 port switch/router implementing the Intel® QuickPath Interconnect Link and Routing layers. The R-Box is responsible for routing and transmitting all intra- and inter-processor communication.

The on-die agents include two B-Boxes (ports 4/7), two S-boxes (ports 2/6) and the U-Box (which shares a connection with B-Box1 on Port7). The R-Box connects to these through full flit 80b links. Ports 0,1,4 and 5 are connected to external Intel QPI agents (through P-boxes also known as the physical layers), also through full flit 80b links.

The R-Box consists of 8 identical ports and a wire crossbar that connects the ports together. Each port contains three main sections as shown in the following figure: the input port, the output port, and the arbitration control.

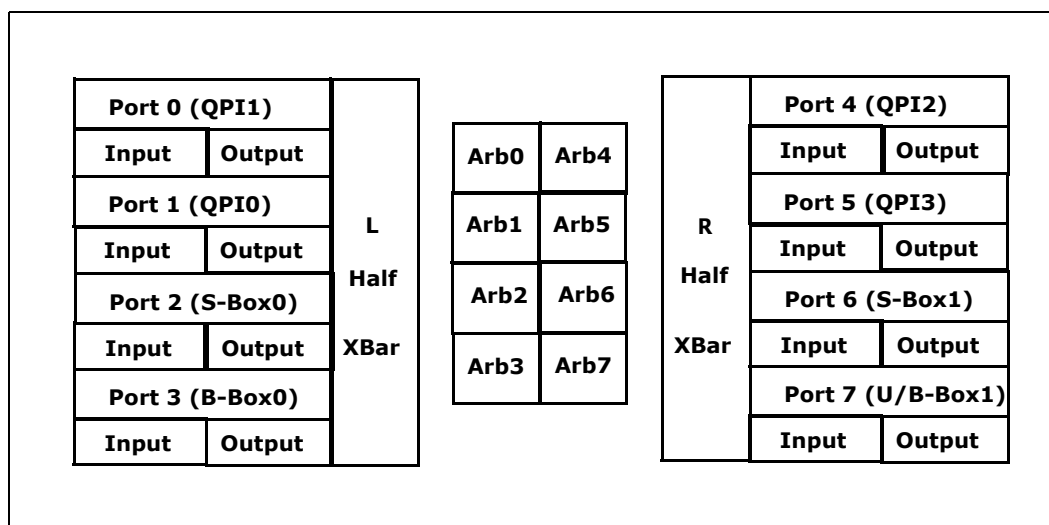


Figure 2-1. R-Box Block Diagram

2.6.1.1 R-Box Input Port

The R-Box input port is responsible for storing incoming packets from the B and S-Boxes as well as off-chip requests using the Intel® QuickPath Interconnect protocol. Data from each packet header is consolidated and sent to the R-Box arbiter.

R-Box input ports have two structures of important to performance monitoring; Entry overflow table (EOT) and Entry Table (ET). R-Box PMU supports performance monitoring in these two structures.

2.6.1.2 R-Box Arbitration Control

The R-Box arbitration control is responsible for selecting when packets move from the input ports to the output ports and which output port they go to if there are multiple options specified.

R-Box arbitration does not have any storage structures. This part of the logic basically determines which port to route the packet and then arbitrate to secure a route to that port through the cross-bar.

The arbitration is done at 3 levels: queue, port and global arbitration. R-Box PMUs support performance monitoring at the arbitration control.

2.6.1.3 R-Box Output Port

The R-Box output port acts as a virtual wire that is responsible for de-coupling the crossbar from further downstream paths to on-chip or off-chip ports while carrying out the Link layer functions.

2.6.1.4 R-Box Link Layer Resources

Each R-Box port supports up to three virtual networks (VN0, VN1, and VNA) as defined by the *Intel® QuickPath Interconnect Specification*. The following table specifies the port resources.

Table 2-38. Input Buffering Per Port

| Message Class | Abbr | VNA | | VN0 | | VN1 | |
|-----------------------|------|------|-------|------|----------|------|----------|
| | | Pkts | Flits | Pkts | Flits | Pkts | Flits |
| Home | HOM | | 96 | 1 | 1 | 1 | 1 |
| Snoop | SNP | | | 1 | 1 | 1 | 1 |
| Non-Data Response | NDR | | | 1 | 1 | 1 | 1 |
| Data Response | DRS | | | 1 | up to 11 | 1 | up to 11 |
| Non-Coherent Standard | NCS | | | 1 | up to 3 | 1 | up to 3 |
| Non-Coherent Bypass | NCB | | | 1 | up to 11 | 1 | up to 11 |

2.6.2 R-Box Performance Monitoring Overview

The R-Box supports performance event monitoring through its Performance Monitoring Unit (PMU). At a high level, the R-Box PMU supports features comparable to other uncore PMUs. R-Box PMUs support both Global and Local PMU freeze/unfreeze. R-Box PMUs are accessible through Machine Specific Registers (MSRs). R-Box PMU consists of 16 48b-wide performance monitoring data counters and a collection of other peripheral control registers.

For information on how to setup a monitoring session, refer to [Section 2.1.3, "Setting up a Monitoring Session"](#).

The counters, along with the control register paired with each one, are split. Half of the counters (0-7) can monitor events occurring on the 'left' side of the R-Box (ports 0-3) and the other half (8-15) monitor ports 4-7 on the 'right' side.

Since the R-Box consists of 12 almost identical ports, R-Box perfmon events consist of an identical set of events for each port. The R-Box perfmon usage model allows monitoring of multiple ports at the same time. R-Box PMUs do not provide any global performance monitoring events.

However, unlike many other uncore boxes, event programming in the R-Box is hierarchical. It is necessary to program multiple MSRs to select the event to be monitored. In order to program an event, each of the control registers for its accompanying counter must be redirected to a subcontrol register

attached to a specific port. Each control register can be redirected to one of 2 IPERF control registers (for RIX events), one of 2 fields in a QLX control register or one of 2 mask/match registers. Therefore, it is possible to monitor up to two of any event per port.

The R-Box also includes a pair of mask/match registers on each port that allow a user to match packets serviced (packet is transferred from input to output port) by the R-Box according to various standard packet fields such as message class, opcode, etc.

2.6.2.1 Choosing An Event To Monitor - Example

- 1) Pick an event to monitor (e.g. FLITS_SENT)
- 2) Pick a port to monitor on (e.g. QPI0)
- 3) Pick a generic counter (control+data) that can monitor an event on that port. (e.g R_MSR_PMON_CTL/CTR3)
- 4) Pick one of the two sub counters that allows a user to monitor the event (R_MSR_PORT1_IPERF1), program it to monitor the chosen event (R_MSR_PORT1_IPERF1[31] = 0x1) and set the generic control to point to it (R_MSR_PMON_CTL3.ev_sel == 0x7).
- 5) Enable the counter (e.g. R_MSR_PMON_CTL3.en == 0x1)

2.6.2.2 R-Box PMU - Overflow, Freeze and Unfreeze

If an overflow is detected from a R-Box performance counter, the overflow bit is set at the box level (R_MSR_PMON_GLOBAL_STATUS_15_8.ov for the R side and R_MSR_PMON_GLOBAL_STATUS_7_0.ov for the L), and forwarded up the chain towards the U-Box. If counter overflows in the left R-Box, a notification is sent and stored in S-Box0 (S_MSR_PMON_SUMMARY.ov_r) which, in turn, sends the overflow notification up to the U-Box (U_MSR_PMON_GLOBAL_STATUS.ov_s0). Refer to [Table 2-26, "S_MSR_PMON_SUMMARY Register Fields"](#) to determine how each R-Box's overflow bit is accumulated in the attached S-Box.

HW can be also configured (by setting the corresponding *.pmi_en* to 1) to send a PMI to the U-Box when an overflow is detected. The U-Box may be configured to freeze all uncore counting and/or send a PMI to selected cores when it receives this signal.

Once a freeze has occurred, in order to see a new freeze, the overflow field responsible for the freeze, must be cleared by setting the corresponding bit in R_MSR_PMON_GLOBAL_OVF_CTL.clr_ov. Assuming all the counters have been locally enabled (*.en* bit in data registers meant to monitor events) and the overflow bit(s) has been cleared, the R-Box is prepared for a new sample interval. Once the global controls have been re-enabled ([Section 2.1.5, "Enabling a New Sample Interval from Frozen Counters"](#)), counting will resume.

2.6.3 R-BOX Performance Monitors

Table 2-39. R-Box Performance Monitoring MSRs

| MSR Name | Access | MSR Address | Size (bits) | Description |
|-----------------------------|--------|-------------|-------------|----------------------------------|
| R_MSR_PORT7_XBR_SET2_MASK | RW_NA | 0x0E9E | 64 | R-Box Port 7 Mask 2 |
| R_MSR_PORT7_XBR_SET2_MATCH | RW_NA | 0x0E9D | 64 | R-Box Port 7 Match 2 |
| R_MSR_PORT7_XBR_SET2_MM_CFG | RW_NA | 0x0E9C | 64 | R-Box Port 7 Mask/Match Config 2 |
| R_MSR_PORT7_XBR_SET1_MASK | RW_NA | 0x0E8E | 64 | R-Box Port 7 Mask 1 |
| R_MSR_PORT7_XBR_SET1_MATCH | RW_NA | 0x0E8D | 64 | R-Box Port 7 Match 1 |
| R_MSR_PORT7_XBR_SET1_MM_CFG | RW_NA | 0x0E8C | 64 | R-Box Port 7 Mask/Match Config 1 |
| | | | | |

| MSR Name | Access | MSR Address | Size (bits) | Description |
|-----------------------------|--------|-------------|-------------|----------------------------------|
| R_MSR_PORT6_XBR_SET2_MASK | RW_NA | 0x0E9A | 64 | R-Box Port 6 Mask 2 |
| R_MSR_PORT6_XBR_SET2_MATCH | RW_NA | 0x0E99 | 64 | R-Box Port 6 Match 2 |
| R_MSR_PORT6_XBR_SET2_MM_CFG | RW_NA | 0x0E98 | 64 | R-Box Port 6 Mask/Match Config 2 |
| R_MSR_PORT6_XBR_SET1_MASK | RW_NA | 0x0E8A | 64 | R-Box Port 6 Mask 1 |
| R_MSR_PORT6_XBR_SET1_MATCH | RW_NA | 0x0E89 | 64 | R-Box Port 6 Match 1 |
| R_MSR_PORT6_XBR_SET1_MM_CFG | RW_NA | 0x0E88 | 64 | R-Box Port 6 Mask/Match Config 1 |
| | | | | |
| R_MSR_PORT5_XBR_SET2_MASK | RW_NA | 0x0E96 | 64 | R-Box Port 5 Mask 2 |
| R_MSR_PORT5_XBR_SET2_MATCH | RW_NA | 0x0E95 | 64 | R-Box Port 5 Match 2 |
| R_MSR_PORT5_XBR_SET2_MM_CFG | RW_NA | 0x0E94 | 64 | R-Box Port 5 Mask/Match Config 2 |
| R_MSR_PORT5_XBR_SET1_MASK | RW_NA | 0x0E86 | 64 | R-Box Port 5 Mask 1 |
| R_MSR_PORT5_XBR_SET1_MATCH | RW_NA | 0x0E85 | 64 | R-Box Port 5 Match 1 |
| R_MSR_PORT5_XBR_SET1_MM_CFG | RW_NA | 0x0E84 | 64 | R-Box Port 5 Mask/Match Config 1 |
| | | | | |
| R_MSR_PORT4_XBR_SET2_MASK | RW_NA | 0x0E92 | 64 | R-Box Port 4 Mask 2 |
| R_MSR_PORT4_XBR_SET2_MATCH | RW_NA | 0x0E91 | 64 | R-Box Port 4 Match 2 |
| R_MSR_PORT4_XBR_SET2_MM_CFG | RW_NA | 0x0E90 | 64 | R-Box Port 4 Mask/Match Config 2 |
| R_MSR_PORT4_XBR_SET1_MASK | RW_NA | 0x0E82 | 64 | R-Box Port 4 Mask 1 |
| R_MSR_PORT4_XBR_SET1_MATCH | RW_NA | 0x0E81 | 64 | R-Box Port 4 Match 1 |
| R_MSR_PORT4_XBR_SET1_MM_CFG | RW_NA | 0x0E80 | 64 | R-Box Port 4 Mask/Match Config 1 |
| | | | | |
| R_MSR_PORT3_XBR_SET2_MASK | RW_NA | 0x0E7E | 64 | R-Box Port 3 Mask 2 |
| R_MSR_PORT3_XBR_SET2_MATCH | RW_NA | 0x0E7D | 64 | R-Box Port 3 Match 2 |
| R_MSR_PORT3_XBR_SET2_MM_CFG | RW_NA | 0x0E7C | 64 | R-Box Port 3 Mask/Match Config 2 |
| R_MSR_PORT3_XBR_SET1_MASK | RW_NA | 0x0E6E | 64 | R-Box Port 3 Mask 1 |
| R_MSR_PORT3_XBR_SET1_MATCH | RW_NA | 0x0E6D | 64 | R-Box Port 3 Match 1 |
| R_MSR_PORT3_XBR_SET1_MM_CFG | RW_NA | 0x0E6C | 64 | R-Box Port 3 Mask/Match Config 1 |
| | | | | |
| R_MSR_PORT2_XBR_SET2_MASK | RW_NA | 0x0E7A | 64 | R-Box Port 2 Mask 2 |
| R_MSR_PORT2_XBR_SET2_MATCH | RW_NA | 0x0E79 | 64 | R-Box Port 2 Match 2 |
| R_MSR_PORT2_XBR_SET2_MM_CFG | RW_NA | 0x0E78 | 64 | R-Box Port 2 Mask/Match Config 2 |
| R_MSR_PORT2_XBR_SET1_MASK | RW_NA | 0x0E6A | 64 | R-Box Port 2 Mask 1 |
| R_MSR_PORT2_XBR_SET1_MATCH | RW_NA | 0x0E69 | 64 | R-Box Port 2 Match 1 |
| R_MSR_PORT2_XBR_SET1_MM_CFG | RW_NA | 0x0E68 | 64 | R-Box Port 2 Mask/Match Config 1 |
| | | | | |
| R_MSR_PORT1_XBR_SET2_MASK | RW_NA | 0x0E76 | 64 | R-Box Port 1 Mask 2 |
| R_MSR_PORT1_XBR_SET2_MATCH | RW_NA | 0x0E75 | 64 | R-Box Port 1 Match 2 |
| R_MSR_PORT1_XBR_SET2_MM_CFG | RW_NA | 0x0E74 | 64 | R-Box Port 1 Mask/Match Config 2 |
| R_MSR_PORT1_XBR_SET1_MASK | RW_NA | 0x0E66 | 64 | R-Box Port 1 Mask 1 |
| R_MSR_PORT1_XBR_SET1_MATCH | RW_NA | 0x0E65 | 64 | R-Box Port 1 Match 1 |
| R_MSR_PORT1_XBR_SET1_MM_CFG | RW_NA | 0x0E64 | 64 | R-Box Port 1 Mask/Match Config 1 |

| MSR Name | Access | MSR Address | Size (bits) | Description |
|-------------------------------|---------|-------------|-------------|--|
| R_MSR_PORT0_XBR_SET2_MASK | RW_NA | 0x0E72 | 64 | R-Box Port 0 Mask 2 |
| R_MSR_PORT0_XBR_SET2_MATCH | RW_NA | 0x0E71 | 64 | R-Box Port 0 Match 2 |
| R_MSR_PORT0_XBR_SET2_MM_CFG | RW_NA | 0x0E70 | 64 | R-Box Port 0 Mask/Match Config 2 |
| R_MSR_PORT0_XBR_SET1_MASK | RW_NA | 0x0E62 | 64 | R-Box Port 0 Mask 1 |
| R_MSR_PORT0_XBR_SET1_MATCH | RW_NA | 0x0E61 | 64 | R-Box Port 0 Match 1 |
| R_MSR_PORT0_XBR_SET1_MM_CFG | RW_NA | 0x0E60 | 64 | R-Box Port 0 Mask/Match Config 1 |
| R_MSR_PMON_CTR15 | RW_RW | 0x0E3F | 64 | R-Box PMON Counter 15 |
| R_MSR_PMON_CTL15 | RW_NA | 0x0E3E | 64 | R-Box PMON Control 15 |
| R_MSR_PMON_CTR14 | RW_RW | 0x0E3D | 64 | R-Box PMON Counter 14 |
| R_MSR_PMON_CTL14 | RW_NA | 0x0E3C | 64 | R-Box PMON Control 14 |
| R_MSR_PMON_CTR13 | RW_RW | 0x0E3B | 64 | R-Box PMON Counter 13 |
| R_MSR_PMON_CTL13 | RW_NA | 0x0E3A | 64 | R-Box PMON Control 13 |
| R_MSR_PMON_CTR12 | RW_RW | 0x0E39 | 64 | R-Box PMON Counter 12 |
| R_MSR_PMON_CTL12 | RW_NA | 0x0E38 | 64 | R-Box PMON Control 12 |
| R_MSR_PMON_CTR11 | RW_RW | 0x0E37 | 64 | R-Box PMON Counter 11 |
| R_MSR_PMON_CTL11 | RW_NA | 0x0E36 | 64 | R-Box PMON Control 11 |
| R_MSR_PMON_CTR10 | RW_RW | 0x0E35 | 64 | R-Box PMON Counter 10 |
| R_MSR_PMON_CTL10 | RW_NA | 0x0E34 | 64 | R-Box PMON Control 10 |
| R_MSR_PMON_CTR9 | RW_RW | 0x0E33 | 64 | R-Box PMON Counter 9 |
| R_MSR_PMON_CTL9 | RW_NA | 0x0E32 | 64 | R-Box PMON Control 9 |
| R_MSR_PMON_CTR8 | RW_RW | 0x0E31 | 64 | R-Box PMON Counter 8 |
| R_MSR_PMON_CTL8 | RW_NA | 0x0E30 | 64 | R-Box PMON Control 8 |
| R_MSR_PORT7_QLX_CFG | RW_NA | 0x0E2F | 32 | R-Box Port 7 QLX Perf Event Cfg |
| R_MSR_PORT6_QLX_CFG | RW_NA | 0x0E2E | 32 | R-Box Port 6 QLX Perf Event Cfg |
| R_MSR_PORT5_QLX_CFG | RW_NA | 0x0E2D | 32 | R-Box Port 5 QLX Perf Event Cfg |
| R_MSR_PORT4_QLX_CFG | RW_NA | 0x0E2C | 32 | R-Box Port 4 QLX Perf Event Cfg |
| R_MSR_PORT7_IPERF_CFG1 | RW_NA | 0x0E2B | 32 | R-Box Port 7 RIX Perf Event Cfg 1 |
| R_MSR_PORT6_IPERF_CFG1 | RW_NA | 0x0E2A | 32 | R-Box Port 6 RIX Perf Event Cfg 1 |
| R_MSR_PORT5_IPERF_CFG1 | RW_NA | 0x0E29 | 32 | R-Box Port 5 RIX Perf Event Cfg 1 |
| R_MSR_PORT4_IPERF_CFG1 | RW_NA | 0x0E28 | 32 | R-Box Port 4 RIX Perf Event Cfg 1 |
| R_MSR_PORT3_IPERF_CFG1 | RW_NA | 0x0E27 | 32 | R-Box Port 3 RIX Perf Event Cfg 1 |
| R_MSR_PORT2_IPERF_CFG1 | RW_NA | 0x0E26 | 32 | R-Box Port 2 RIX Perf Event Cfg 1 |
| R_MSR_PORT1_IPERF_CFG1 | RW_NA | 0x0E25 | 32 | R-Box Port 1 RIX Perf Event Cfg 1 |
| R_MSR_PORT0_IPERF_CFG1 | RW_NA | 0x0E24 | 32 | R-Box Port 0 RIX Perf Event Cfg 1 |
| R_MSR_PMON_OVF_CTL_15_8 | RW1C_WO | 0x0E22 | 32 | R-Box PMON Overflow Ctrl for ctrs 15:8 |
| R_MSR_PMON_GLOBAL_STATUS_15_8 | RO_WO | 0x0E21 | 32 | R-Box PMON Global Status for ctrs 15:8 |

| MSR Name | Access | MSR Address | Size (bits) | Description |
|------------------------------|---------|-------------|-------------|---|
| R_MSR_PMON_GLOBAL_CTL_15_8 | RW_NA | 0x0E20 | 32 | R-Box PMON Global Control Counters 15:8 |
| R_MSR_PMON_CTR7 | RW_RW | 0x0E1F | 64 | R-Box PMON Counter 7 |
| R_MSR_PMON_CTL7 | RW_NA | 0x0E1E | 64 | R-Box PMON Control 7 |
| R_MSR_PMON_CTR6 | RW_RW | 0x0E1D | 64 | R-Box PMON Counter 6 |
| R_MSR_PMON_CTL6 | RW_NA | 0x0E1C | 64 | R-Box PMON Control 6 |
| R_MSR_PMON_CTR5 | RW_RW | 0x0E1B | 64 | R-Box PMON Counter 5 |
| R_MSR_PMON_CTL5 | RW_NA | 0x0E1A | 64 | R-Box PMON Control 5 |
| R_MSR_PMON_CTR4 | RW_RW | 0x0E19 | 64 | R-Box PMON Counter 4 |
| R_MSR_PMON_CTL4 | RW_NA | 0x0E18 | 64 | R-Box PMON Control 4 |
| R_MSR_PMON_CTR3 | RW_RW | 0x0E17 | 64 | R-Box PMON Counter 3 |
| R_MSR_PMON_CTL3 | RW_NA | 0x0E16 | 64 | R-Box PMON Control 3 |
| R_MSR_PMON_CTR2 | RW_RW | 0x0E15 | 64 | R-Box PMON Counter 2 |
| R_MSR_PMON_CTL2 | RW_NA | 0x0E14 | 64 | R-Box PMON Control 2 |
| R_MSR_PMON_CTR1 | RW_RW | 0x0E13 | 64 | R-Box PMON Counter 1 |
| R_MSR_PMON_CTL1 | RW_NA | 0x0E12 | 64 | R-Box PMON Control 1 |
| R_MSR_PMON_CTR0 | RW_RW | 0x0E11 | 64 | R-Box PMON Counter 0 |
| R_MSR_PMON_CTL0 | RW_NA | 0x0E10 | 64 | R-Box PMON Control 0 |
| R_MSR_PORT3_QLX_CFG | RW_NA | 0x0E0F | 32 | R-Box Port 3 QLX Perf Event Cfg |
| R_MSR_PORT2_QLX_CFG | RW_NA | 0x0E0E | 32 | R-Box Port 2 QLX Perf Event Cfg |
| R_MSR_PORT1_QLX_CFG | RW_NA | 0x0E0D | 32 | R-Box Port 1 QLX Perf Event Cfg |
| R_MSR_PORT0_QLX_CFG | RW_NA | 0x0E0C | 32 | R-Box Port 0 QLX Perf Event Cfg |
| R_MSR_PORT7_IPERF_CFG0 | RW_NA | 0x0E0B | 32 | R-Box Port 7 RIX Perf Event Cfg 0 |
| R_MSR_PORT6_IPERF_CFG0 | RW_NA | 0x0E0A | 32 | R-Box Port 6 RIX Perf Event Cfg 0 |
| R_MSR_PORT5_IPERF_CFG0 | RW_NA | 0x0E09 | 32 | R-Box Port 5 RIX Perf Event Cfg 0 |
| R_MSR_PORT4_IPERF_CFG0 | RW_NA | 0x0E08 | 32 | R-Box Port 4 RIX Perf Event Cfg 0 |
| R_MSR_PORT3_IPERF_CFG0 | RW_NA | 0x0E07 | 32 | R-Box Port 3 RIX Perf Event Cfg 0 |
| R_MSR_PORT2_IPERF_CFG0 | RW_NA | 0x0E06 | 32 | R-Box Port 2 RIX Perf Event Cfg 0 |
| R_MSR_PORT1_IPERF_CFG0 | RW_NA | 0x0E05 | 32 | R-Box Port 1 RIX Perf Event Cfg 0 |
| R_MSR_PORT0_IPERF_CFG0 | RW_NA | 0x0E04 | 32 | R-Box Port 0 RIX Perf Event Cfg 0 |
| R_MSR_PMON_OVF_CTL_15_8 | RW1C_WO | 0x0E02 | 32 | R-Box PMON Overflow Ctrl for ctrs 7:0 |
| R_MSR_PMON_GLOBAL_STATUS_7_0 | RO_WO | 0x0E01 | 32 | R-Box PMON Global Status for ctrs 7:0 |
| R_MSR_PMON_GLOBAL_CTL_7_0 | RW_NA | 0x0E00 | 32 | R-Box PMON Global Control Counters 7:0 |

2.6.3.1 R-Box Performance Monitors To Port Mapping

Table 2-40. R-Box Port Map

| Port ID | R-Box Port# | L/R Box | PMU Cnt 0-7 | PMU Cnt 8-15 | IPERF Addresses | ARB_PERF Addresses | Match/Mask Addresses |
|----------|-------------|---------|-------------|--------------|-----------------|--------------------|--------------------------|
| QPI1 | 0 | L | b000 | NA | 0xE24,0xE04 | 0xE0C | 0xE72-0xE70, 0xE62-0xE60 |
| QPI0 | 1 | L | b001 | NA | 0xE25,0xE05 | 0xE0D | 0xE76-0xE74, 0xE66-0xE64 |
| S-Box0 | 2 | L | b010 | NA | 0xE26,0xE06 | 0xE0E | 0xE7A-0xE78, 0xE6A-0xE68 |
| B-Box0 | 3 | L | b011 | NA | 0xE27,0xE07 | 0xE0F | 0xE7E-0xE7C, 0xE6E-0xE6C |
| QPI2 | 4 | R | NA | b100 | 0xE28,0xE08 | 0xE2C | 0xE92-0xE90, 0xE82-0xE80 |
| QPI3 | 5 | R | NA | b000 | 0xE29,0xE09 | 0xE2D | 0xE96-0xE94, 0xE86-0xE84 |
| S-Box1 | 6 | R | NA | b001 | 0xE2A,0xE0A | 0xE2E | 0xE9A-0xE98, 0xE8A-0xE88 |
| U/B-Box1 | 7 | R | NA | b010 | 0xE2B,0xE0B | 0xE2F | 0xE9E-0xE9C, 0xE8E-0xE8C |

2.6.3.2 R-Box Box Level PMON state

The following registers represent the state governing all box-level PMUs in the R-Box.

The `_GLOBAL_CTL` register contains the bits used to enable monitoring. It is necessary to set the `.ctr_en` bit to 1 before the corresponding data register can collect events.

If an overflow is detected from one of the R-Box PMON registers, the corresponding bit in the `_GLOBAL_STATUS.ov` field will be set. To reset the overflow bits set in the `_GLOBAL_STATUS.ov` field, a user must set the corresponding bits in the `_GLOBAL_OVF_CTL.clr_ov` field before beginning a new sample interval.

Table 2-41. R_MSR_PMON_GLOBAL_CTL_{15_8, 7_0} Register Fields

| Field | Bits | HW Reset Value | Description |
|---------------------|------|----------------|--|
| <code>ctr_en</code> | 7:0 | 0 | Must be set to enable each RBOX counter (bit 0 to enable ctr0, etc) NOTE: U-Box enable and per counter enable must also be set to fully enable the counter. |

Table 2-42. R_MSR_PMON_GLOBAL_STATUS_{15_8, 7_0} Register Fields

| Field | Bits | HW Reset Value | Description |
|-----------------|------|----------------|--|
| <code>ov</code> | 7:0 | 0 | If an overflow is detected from the corresponding RBOX PMON register, it's overflow bit will be set. |

Table 2-43. R_MSR_PMON_OVF_CTL_{15_8, 7_0} Register Fields

| Field | Bits | HW Reset Value | Description |
|--------|------|----------------|--|
| clr_ov | 7:0 | 0 | Writing bit in field to '1' will clear the corresponding overflow bit in R_CSR_PMON_GLOBAL_STATUS_{15_8,7_0} to 0. |

2.6.3.3 R-Box PMON state - Counter/Control Pairs + Filters

The following table defines the layout of the R-Box performance monitor control registers. The main task of these configuration registers is to select the subcontrol register that selects the event to be monitored by the respective data counter. Setting the `.ev_sel/` fields performs the subcontrol register selection. The `.en` bit must be set to 1 to enable counting.

Additional control bits include:

- `.pmi_en` governs what to do if an overflow is detected.

Table 2-44. R_MSR_PMON_CTL{15-0} Register – Field Definitions

| Field | Bits | HW Reset Value | Description |
|--------|-------|----------------|---|
| ig | 63 | 0 | Read zero; writes ignored. |
| rsv | 62:61 | 0 | Reserved; Must write to 0 else behavior is undefined. |
| ig | 60:7 | 0 | Read zero; writes ignored. |
| pmi_en | 6 | 0 | When this bit is asserted and the corresponding counter overflows, a PMI exception is sent to the U-Box. |
| ev_sel | 5:1 | 0 | Event Select For the R-Box this means choosing which sub register contains the actual event select. Each control register can redirect the event select to one of 3 sets of registers: QLX, RIX or Mask/Match registers. It can further select from one of two subselect fields (either in the same or different registers). And finally, each control can 'listen' to events occurring on one of 4 ports. The first 8 control registers can refer to the first 4 ports and the last 8 control So, for example: RP_CR_R_CSR_PMON_CTL9 can refer to R_CSR_PORT{4-7}_IPERF{0-1} |
| en | 0 | 0 | Local Counter Enable. When set, the associated counter is locally enabled. NOTE: It must also be enabled in the governing R_MSR_PMON_GLOBAL_CTL register and the U-Box to be fully enabled. |

Table 2-45. R_MSR_PMON_CTL{15-8} Event Select

| Name | Code | Description |
|----------------|-----------|--|
| PORT4_IPERF0 | 0x00 | Select Event Configured in R_CSR_PORT4_IPERF0 |
| PORT4_IPERF1 | 0x01 | Select Event Configured in R_CSR_PORT4_IPERF1 |
| PORT4_QLX0 | 0x02 | Select Event Configured in R_CSR_PORT4_QLX_EVENT_CFG[*0] |
| PORT4_QLX1 | 0x03 | Select Event Configured in R_CSR_PORT4_QLX_EVENT_CFG[*1] |
| PORT4_XBAR_MM1 | 0x04 | Set1 Port4 XBAR Mask/Match |
| PORT4_XBAR_MM2 | 0x05 | Set2 Port4 XBAR Mask/Match |
| PORT5_IPERF0 | 0x06 | Select Event Configured in R_CSR_PORT5_IPERF0 |
| PORT5_IPERF1 | 0x07 | Select Event Configured in R_CSR_PORT5_IPERF1 |
| PORT5_QLX0 | 0x08 | Select Event Configured in R_CSR_PORT5_QLX_EVENT_CFG[*0] |
| PORT5_QLX1 | 0x09 | Select Event Configured in R_CSR_PORT5_QLX_EVENT_CFG[*1] |
| PORT5_XBAR_MM1 | 0x0A | Set1 Port5 XBAR Mask/Match |
| PORT5_XBAR_MM2 | 0x0B | Set2 Port5 XBAR Mask/Match |
| PORT6_IPERF0 | 0x0C | Select Event Configured in R_CSR_PORT6_IPERF0 |
| PORT6_IPERF1 | 0x0D | Select Event Configured in R_CSR_PORT6_IPERF1 |
| PORT6_QLX0 | 0x0E | Select Event Configured in R_CSR_PORT6_QLX_EVENT_CFG[*0] |
| PORT6_QLX1 | 0x0F | Select Event Configured in R_CSR_PORT6_QLX_EVENT_CFG[*1] |
| PORT6_XBAR_MM1 | 0x10 | Set1 Port6 XBAR Mask/Match |
| PORT6_XBAR_MM2 | 0x11 | Set2 Port6 XBAR Mask/Match |
| PORT7_IPERF0 | 0x12 | Select Event Configured in R_CSR_PORT7_IPERF0 |
| PORT7_IPERF1 | 0x13 | Select Event Configured in R_CSR_PORT7_IPERF1 |
| PORT7_QLX0 | 0x14 | Select Event Configured in R_CSR_PORT7_QLX_EVENT_CFG[*0] |
| PORT7_QLX1 | 0x15 | Select Event Configured in R_CSR_PORT7_QLX_EVENT_CFG[*1] |
| PORT7_XBAR_MM1 | 0x16 | Set1 Port7 XBAR Mask/Match |
| PORT7_XBAR_MM2 | 0x17 | Set2 Port7 XBAR Mask/Match |
| ILLEGAL | 0x18-0x1F | (* illegal selection *) |

Table 2-46. R_MSR_PMON_CTL{7-0} Event Select

| Name | Code | Description |
|----------------|-----------|--|
| PORT0_IPERF0 | 0x00 | Select Event Configured in R_CSR_PORT0_IPERF0 |
| PORT0_IPERF1 | 0x01 | Select Event Configured in R_CSR_PORT0_IPERF1 |
| PORT0_QLX0 | 0x02 | Select Event Configured in R_CSR_PORT0_QLX_EVENT_CFG[*0] |
| PORT0_QLX1 | 0x03 | Select Event Configured in R_CSR_PORT0_QLX_EVENT_CFG[*1] |
| PORT0_XBAR_MM1 | 0x04 | Set1 Port0 XBAR Mask/Match |
| PORT0_XBAR_MM2 | 0x05 | Set2 Port0 XBAR Mask/Match |
| PORT1_IPERF0 | 0x06 | Select Event Configured in R_CSR_PORT1_IPERF0 |
| PORT1_IPERF1 | 0x07 | Select Event Configured in R_CSR_PORT1_IPERF1 |
| PORT1_QLX0 | 0x08 | Select Event Configured in R_CSR_PORT1_QLX_EVENT_CFG[*0] |
| PORT1_QLX1 | 0x09 | Select Event Configured in R_CSR_PORT1_QLX_EVENT_CFG[*1] |
| PORT1_XBAR_MM1 | 0x0A | Set1 Port1 XBAR Mask/Match |
| PORT1_XBAR_MM2 | 0x0B | Set2 Port1 XBAR Mask/Match |
| PORT2_IPERF0 | 0x0C | Select Event Configured in R_CSR_PORT2_IPERF0 |
| PORT2_IPERF1 | 0x0D | Select Event Configured in R_CSR_PORT2_IPERF1 |
| PORT2_QLX0 | 0x0E | Select Event Configured in R_CSR_PORT2_QLX_EVENT_CFG[*0] |
| PORT2_QLX1 | 0x0F | Select Event Configured in R_CSR_PORT2_QLX_EVENT_CFG[*1] |
| PORT2_XBAR_MM1 | 0x10 | Set1 Port2 XBAR Mask/Match |
| PORT2_XBAR_MM2 | 0x11 | Set2 Port2 XBAR Mask/Match |
| PORT3_IPERF0 | 0x12 | Select Event Configured in R_CSR_PORT3_IPERF0 |
| PORT3_IPERF1 | 0x13 | Select Event Configured in R_CSR_PORT3_IPERF1 |
| PORT3_QLX0 | 0x14 | Select Event Configured in R_CSR_PORT3_QLX_EVENT_CFG[*0] |
| PORT3_QLX1 | 0x15 | Select Event Configured in R_CSR_PORT3_QLX_EVENT_CFG[*1] |
| PORT3_XBAR_MM1 | 0x16 | Set1 Port3 XBAR Mask/Match |
| PORT3_XBAR_MM2 | 0x17 | Set2 Port3 XBAR Mask/Match |
| ILLEGAL | 0x18-0x1F | (* illegal selection *) |

The R-Box performance monitor data registers are 48b wide. A counter overflow occurs when a carry out bit from bit 47 is detected. Software can force all uncore counting to freeze after N events by preloading a monitor with a count value of $2^{48} - N$ and setting the control register to send a PMI to the U-Box. Upon receipt of the PMI, the U-Box will disable counting ([Section 2.1.1, "Freezing on Counter Overflow"](#)). During the interval of time between overflow and global disable, the counter value will wrap and continue to collect events.

In this way, software can capture the precise number of events that occurred between the time uncore counting was enabled and when it was disabled (or 'frozen') with minimal skew.

If accessible, software can continuously read the data registers without disabling event collection.

Table 2-47. R_MSR_PMON_CTR{15-0} Register – Field Definitions

| Field | Bits | HW Reset Value | Description |
|-------------|------|----------------|----------------------------------|
| event_count | 47:0 | 0 | 48-bit performance event counter |

2.6.3.4 R-Box IPERF Performance Monitoring Control Registers

The following table contains the events that can be monitored if one of the RIX (IPERF) registers was chosen to select the event.

Table 2-48. R_MSR_PORT{7-0}_IPERF_CFG{1-0} Registers(Sheet 1 of 2)

| Field | Bits | HW Reset Value | Description |
|---------------|-------|----------------|--|
| FLT_SENT | 31 | 0x0 | Flit Sent |
| NULL_IDLE | 30 | 0x0 | Null Idle Flit Sent |
| RETRYQ_OV | 29 | 0x0 | Retry Queue Overflowed in this Output Port. |
| RETRYQ_NE | 28 | 0x0 | Retry Queue Not Empty in this Output Port |
| OUTQ_OV | 27 | 0x0 | Output Queue Overflowed in this Output Port |
| OUTQ_NE | 26 | 0x0 | Output Queue Not Empty in this Output Port |
| RCVD_SPEC_FLT | 25 | 0x0 | Special Flit Received |
| RCVD_ERR_FLT | 24 | 0x0 | Flit Received which caused CRC Error |
| ig | 23:22 | 0x0 | Read zero; writes ignored. |
| MC_ROLL_ALLOC | 21 | 0x0 | Used with MC field. If set, every individual allocation of selected MC into EOT is reported. If 0, a 'rolling' count is reported (count whenever count overflows 7b count - val of 128) for selected MC's allocation into EOT. |
| MC | 20:17 | 0x0 | EOT Message Class Count 1000: Data Response - VN1 0111: Non-Coherent Standard 0110: Non-Coherent Bypass 0101: Snoop 0100: Data Response - VNO 0011: Non-Data Response 0010: Home VN1 0001: Home VNO |
| EOT_NE | 16 | 0x0 | Count cycles that EOT is not Empty |
| ARB_SEL | 15:9 | 0x00 | Allocation to Arb Select Bit Mask: 0b1XXXXXX: Home VNO 0bX1XXXXX: Home VN1 0bXX1XXXX: Snoop 0bXXX1XXX: Non-Data Response 0bXXXX1XX: Data Response - VNO/VN1 0bXXXXX1X: Non-Coherent Standard 0bXXXXXX1: Non-Coherent Bypass |

Table 2-48. R_MSR_PORT{7-0}_IPERF_CFG{1-0} Registers(Sheet 2 of 2)

| Field | Bits | HW Reset Value | Description |
|-------------|------|----------------|--|
| IQA_READ_OK | 8 | 0x0 | Bid wins arbitration. Read flit from IQA and drains to XBAR. |
| NEW_PVN | 7:6 | 0x0 | New Packet VN Select: Anded with result of New Packet Class Bit Mask. 11: VNA VN1 VNO 10: VNA 01: VN1 00: VNO |
| NEW_PC | 5:0 | 0x0 | New Packet Class Bit Mask: Bit mask to select which packet types to count. Anded with New Packet VN Select. b1XXXXX: Snoop bX1XXXX: Home bXX1XXX: Non-Data Response bXXX1XX: Data Response bXXXX1X: Non-Coherent Standard bXXXXX1: Non-Coherent Bypass |

2.6.3.5 R-Box QLX Performance Monitoring Control Registers

The following table contains the events that can be monitored if one of the ARB registers was chosen to select the event.

Table 2-49. R_MSR_PORT{7-0}_QLX_CFG Register Fields(Sheet 1 of 2)

| Field | Bits | HW Reset Value | Description |
|---------|-------|----------------|---|
| ig | 31:16 | | Read zero; writes ignored. |
| ev1_sub | 15 | 0x0 | Performance Event 1 Sub-Class Select: 0: VNO 1: VN1 |
| ev1_cls | 14:12 | 0x0 | Performance Event 1 Class Select: 000: HOM 001: SNP 010: NDR 011: NCS 100: DRS 101: NCB 110: VNA - Small 111: VNA - Large |

Table 2-49. R_MSR_PORT{7-0}_QLX_CFG Register Fields(Sheet 2 of 2)

| Field | Bits | HW Reset Value | Description |
|----------|------|----------------|---|
| ev1_type | 11:8 | 0x0 | Performance Event 1 Type Select: 0000: Queue Arb Bid 0001: Local Arb Bid 0010: Global Arb Bid 0011: Queue Arb Fail 0100: Local Arb Fail 0101: Global Arb Fail 0110: Queue Arb Home Order Kill 0111: Local Arb Home Order Kill 1000: Global Arb Home Order Kill 1001: Target Available 1010: Starvation Detected 1011-1111: Reserved |
| ev0_sub | 7 | 0x0 | Performance Event 0Sub-Class Select: 0: VNO 1: VN1 |
| ev0_cls | 6:4 | 0x0 | Performance Event 0 Class Select: 000: HOM 001: SNP 010: NDR 011: NCS 100: DRS 101: NCB 110: VNA - Small 111: VNA - Large |
| ev0_type | 3:0 | 0x0 | Performance Event 0 Type Select: 0000: Queue Arb Bid 0001: Local Arb Bid 0010: Global Arb Bid 0011: Queue Arb Fail 0100: Local Arb Fail 0101: Global Arb Fail 0110: Queue Arb Home Order Kill 0111: Local Arb Home Order Kill 1000: Global Arb Home Order Kill 1001: Target Available 1010: Starvation Detected 1011-1111: Reserved |

2.6.3.6 R-Box Registers for Mask/Match Facility

In addition to generic event counting, each port of the R-Box provides two pairs of MATCH/MASK registers pair that allow a user to filter packet traffic serviced (crossing from an input port to an output port) by the R-Box according to the packet Opcode, Message Class, Response, HNID and Physical Address. Program the selected R-Box counter and IPERF subcounter to capture FLITS to capture the filter match as an event.

To use the match/mask facility :

- a) Set the MM_CFG (see [Table 2-50, "R_MSR_PORT{7-0}_XBR_SET{2-1}_MM_CFG Registers"](#)) *.dis* field (bit 63) to 0 and *.mm_trig_en* (bit 21) to 1.

NOTE: In order to monitor packet traffic, instead of the flit traffic associated with each packet, set `.match_flit_cnt` to 0x1.

b) Program the match/mask regs (see [Table 2-51, "R_MSR_PORT{7-0}_XBR_SET{2-1}_MATCH Registers"](#) and [Table 2-52, "R_MSR_PORT{7-0}_XBR_SET{2-1}_MASK Registers"](#)).

c) Set the counter's control register event select to the appropriate IPERF subcontrol register and set the IPERF register's event select to 0x31 (TO_R_PROG_EV) to capture the mask/match as a performance event.

Table 2-50. R_MSR_PORT{7-0}_XBR_SET{2-1}_MM_CFG Registers

| Field | Bits | HW Reset Value | Description |
|----------------|-------|----------------|--|
| dis | 63 | 0x0 | Disable; Set to 0 to enable use by PMUs. |
| ig | 62:22 | 0x0 | Read zero; writes ignored. |
| mm_trig_en | 21 | 0x0 | Match/Mask trigger enable Set to 1 to enable mask/match trigger |
| ig_flit_cnt | 20 | 0x0 | Ignore flit count Set to ignore match_flit_cnt field |
| match_flit_cnt | 19:16 | 0x0 | Match flit count Set number of flit count in a packet on which to trigger a match event. Ex: Set to '0001' to match on first flit. |
| match_71_64 | 15:8 | 0x0 | upper 8 bits [71:64] of match data |
| mask_71_64 | 7:0 | 0x0 | upper 8 bits [71:64] of mask data |

The following table contains the packet traffic that can be monitored if one of the mask/match registers was chosen to select the event.

Table 2-51. R_MSR_PORT{7-0}_XBR_SET{2-1}_MATCH Registers

| Field | Bits | HW Reset Value | Description |
|-------|-------|----------------|---|
| --- | 63:52 | 0x0 | Reserved; Must write to 0 else behavior is undefined. |
| RDS | 51:48 | 0x0 | Response Data State (valid when MC == DRS and Opcode == 0x0-2). Bit settings are mutually exclusive. b1000 - Modified b0100 - Exclusive b0010 - Shared b0001 - Forwarding b0000 - Invalid (Non-Coherent) |
| --- | 47:36 | 0x0 | Reserved; Must write to 0 else behavior is undefined. |
| RNID | 35:31 | 0x0 | Remote Node ID |
| --- | 30:18 | 0x0 | Reserved; Must write to 0 else behavior is undefined. |
| DNID | 17:13 | 0x0 | Destination Node ID |
| MC | 12:9 | 0x0 | Message Class b0000 HOM - Requests b0001 HOM - Responses b0010 NDR b0011 SNP b0100 NCS --- b1100 NCB --- b1110 DRS |
| OPC | 8:5 | 0x0 | Opcode DRS,NCB: [8] Packet Size, 0 == 9 flits, 1 == 11 flits NCS: [8] Packet Size, 0 == 1 or 2 flits, 1 == 3 flits See Section 2.9, "Packet Matching Reference" for a listing of opcodes that may be filtered per message class. |
| VNW | 4:3 | 0x0 | Virtual Network b00 - VN0 b01 - VN1 b1x - VNA |
| --- | 2:0 | 0x0 | Reserved; Must write to 0 else behavior is undefined. |

Table 2-52. R_MSR_PORT{7-0}_XBR_SET{2-1}_MASK Registers

| Field | Bits | HW Reset Value | Description |
|-------|-------|----------------|---|
| --- | 63:52 | 0x0 | Reserved; Must write to 0 else behavior is undefined. |
| RDS | 51:48 | 0x0 | Response Data State (for certain DRS messages) |
| --- | 47:36 | 0x0 | Reserved; Must write to 0 else behavior is undefined. |
| RNID | 35:31 | 0x0 | Remote Node ID |
| --- | 30:18 | 0x0 | Reserved; Must write to 0 else behavior is undefined. |
| DNID | 17:13 | 0x0 | Destination Node ID |
| MC | 12:9 | 0x0 | Message Class |
| OPC | 8:5 | 0x0 | Opcode See Section 2.9, "Packet Matching Reference" for a listing of opcodes that may be filtered per message class. |
| VNW | 4:3 | 0x0 | Virtual Network |
| --- | 2:0 | 0x0 | Reserved; Must write to 0 else behavior is undefined. |

Following is a selection of common events that may be derived by using the R-Box packet matching facility.

Table 2-53. Message Events Derived from the Match/Mask filters

| Field | Match [15:0] | Mask [15:0] | Description |
|--------------------|---|--|--|
| DRS.AnyDataC | 0x1C00 | 0x1F80 | Any Data Response message containing a cache line in response to a core request. The AnyDataC messages are only sent to an S-Box. The metric DRS.AnyResp - DRS.AnyDataC will compute the number of DRS writeback and non snoop write messages. |
| DRS.DataC_M | 0x1C00 && Match [51:48] 0x8 | 0x1FE0 && Mask [51:48] 0xF | Data Response message of a cache line in M state that is response to a core request. The DRS.DataC_M messages are only sent to S-Boxes. |
| DRS.WbIData | 0x1C80 | 0x1FE0 | Data Response message for Write Back data where cacheline is set to the I state. |
| DRS.WbSData | 0x1CA0 | 0x1FE0 | Data Response message for Write Back data where cacheline is set to the S state. |
| DRS.WbEData | 0x1CC0 | 0x1FE0 | Data Response message for Write Back data where cacheline is set to the E state. |
| DRS.AnyResp | 0x1C00 | 0x1E00 | Any Data Response message. A DRS message can be either 9 flits for a full cache line or 11 flits for partial data. |
| DRS.AnyResp9flits | 0x1C00 | 0x1F00 | Any Data Response message that is 11 flits in length. An 11 flit DRS message contains partial data. Each 8 byte chunk contains an enable field that specifies if the data is valid. |
| DRS.AnyResp11flits | 0x1D00 | 0x1F00 | Any Non Data Response completion message. A NDR message is 1 on flit. |

Table 2-53. Message Events Derived from the Match/Mask filters

| Field | Match [15:0] | Mask [15:0] | Description |
|-------------------|--------------|-------------|---|
| NCB.AnyMsg9flits | 0x1800 | 0x1F00 | Any Non-Coherent Bypass message that is 9 flits in length. A 9 flit NCB message contains a full 64 byte cache line. |
| NCB.AnyMsg11flits | 0x1900 | 0x1F00 | Any Non-Coherent Bypass message that is 11 flits in length. An 11 flit NCB message contains either partial data or an interrupt. For NCB 11 flit data messages, each 8 byte chunk contains an enable field that specifies if the data is valid. |
| NCB.AnyInt | 0x1900 | 0x1F80 | Any Non-Coherent Bypass interrupt message. NCB interrupt messages are 11 flits in length. |

NOTE: Bits 71:16 of the match/mask must be 0 in order to derive these events (except where noted - see DRS.DataC_M). Also the match/mask configuration register should be set to 0x00210000 (bits 21 and 16 set).

2.6.4 R-BOX Performance Monitoring Events

2.6.4.1 An Overview:

The R-Box events provide information on topics such as: a breakdown of traffic as it flows through each of the R-Box's ports (NEW_PACKETS_RECV), raw flit traffic (i.e. FLITS_RECV_ERR or FLITS_SENT), incoming transactions entered into arbitration for outgoing ports (ALLOC_TO_ARB), transactions that fail arbitration (GLOBAL_ARB_BID_FAIL), tracking status of various queues (OUTPUTQ_NE), etc.

In addition, the R-Box provides the ability to match/mask against ALL flit traffic that leaves the R-Box. This is particularly useful for calculating link utilization, throughput and packet traffic broken down by opcode and message class.

2.6.5 R-Box Events Ordered By Code

Table 2-54 summarizes the directly-measured R-Box events.

Table 2-54. Performance Monitor Events for R-Box Events

| Symbol Name | Event Code | Max Inc/ Cyc | Description |
|-------------------|------------|--------------|-------------------------------|
| RIX Events | | IPERF | |
| NEW_PACKETS_RECV | [5:0]0xX | 1 | New Packets Received by Port |
| INQUE_READ_WIN | [8]0x1 | 1 | Input Queue Read Win |
| ALLOC_TO_ARB | [15:9]0xX | 1 | Transactions allocated to Arb |
| EOT_NE_CYCLES | [16]0x1 | 1 | Cycles EOT Not Empty |
| EOT_OCCUPANCY | [21]0x0 | 1 | EOT Occupancy |
| EOT_INSERTS | [21]0x1 | 1 | Number of Inserts into EOT |
| FLITS_RECV_ERR | [24]0x1 | 1 | Error Flits Received |
| FLITS_RECV_SPEC | [25]0x1 | 1 | Special Flits Received |
| OUTPUTQ_NE | [26]0x1 | 1 | Output Queue Not Empty |
| OUTPUTQ_OVFL | [27]0x1 | 1 | Output Queue Overflowed |
| RETRYQ_NE | [28]0x1 | 1 | Retry Queue Not Empty |
| RETRYQ_OV | [29]0x1 | 1 | Retry Queue Overflowed |
| NULL_IDLE | [30]0x1 | 1 | Null Idle Flits |

Table 2-54. Performance Monitor Events for R-Box Events

| Symbol Name | Event Code | Max Inc/ Cyc | Description |
|---------------------|------------|-----------------|------------------------|
| RIX Events | | IPERF | |
| FLITS_SENT | [31]0x1 | 1 | Flits Sent |
| QLX Events | | QLX[3:0] | |
| QUE_ARB_BID | 0x0 | 1 | Queue ARB Bids |
| LOCAL_ARB_BID | 0x1 | 1 | Local ARB Bids |
| GLOBAL_ARB_BID | 0x2 | 1 | Global ARB Bids |
| QUE_ARB_BID_FAIL | 0x3 | 1 | Failed Queue ARB Bids |
| LOCAL_ARB_BID_FAIL | 0x4 | 1 | Failed Local ARB Bids |
| GLOBAL_ARB_BID_FAIL | 0x5 | 1 | Failed Global ARB Bids |
| TARGET_AVAILABLE | 0x9 | 1 | Target Available |
| STARVING | 0xA | 1 | Starvation Detective |

2.6.6 R-Box Performance Monitor Event List

This section enumerates Intel® Xeon® Processor E7 Family uncore performance monitoring events for the R-Box.

ALLOC_TO_ARB

- **Title:** Transactions allocated to ARB
- **Category:** RIX
- **[Bit(s)] Value:** See Note, **Max. Inc/Cyc:** 1,
- **Definition:** Transactions entered into Entry Table (counts incoming messages); This also means they are now available.
- **NOTE:** Any combination of Message Class [15:9] may be monitored.

Table 2-55. Unit Masks for ALLOC_TO_ARB

| Extension | IPERF Bit Values [15:9] | Description |
|-----------|-------------------------|------------------------------------|
| --- | b0000000 | (*nothing will be counted*) |
| NCB | bxxxxx1 | Non-Coherent Bypass Messages |
| NCS | bxxxx1x | Non-Coherent Standard Messages |
| DRS_VN01 | bxxxx1xx | Data Response (VN0 & VN1) Messages |
| NDR | bxxx1xxx | Non-Data Response Messages |
| SNP | bxx1xxxx | Snoop Messages |
| HOM_VN0 | bx1xxxx | Home (VN0) Messages |
| HOM_VN1 | b1xxxxx | Home (VN1) Messages |
| ALL | b1111111 | All Messages |

EOT_INSERTS

- **Title:** Number of Inserts into EOT
- **Category:** RIX
- **[Bit(s)] Value:** [21]0x1, **Max. Inc/Cyc:** 1,
- **Definition:** Used with MC field. Accumulated depth of packets captured in the Entry Overflow Table for specified message types (i.e. EOT count by MC)
- **NOTE:** This basically counts VNA.

Table 2-56. Unit Masks for EOT_DEPTH_ACC

| Extension | IPERF Bit Values [20:17] | Description |
|-----------|--------------------------|--------------------------------|
| --- | b0000 | (*nothing will be counted*) |
| HOM_VN0 | b0001 | Home (VN0) Messages |
| HOM_VN1 | b0010 | Home (VN1) Messages |
| NDR | b0011 | Non-Data Response Messages |
| DRS_VN0 | b0100 | Data Response (VN0) Messages |
| SNP | b0101 | Snoop Messages |
| NCB | b0110 | Non-Coherent Bypass Messages |
| NCS | b0111 | Non-Coherent Standard Messages |
| DRS_VN1 | b1000 | Data Response (VN1) Messages |
| --- | b1001-b1111 | (*nothing will be counted*) |

EOT_NE_CYCLES

- **Title:** Cycles EOT Not Empty
- **Category:** RIX
- **[Bit(s)] Value:** [16]0x1, **Max. Inc/Cyc:** 1,
- **Definition:** Number of cycles the Entry Overflow Table buffer is not empty (MC counts are not all zero).
- **NOTE:** Signals only if EOTs for ALL message classes are empty

EOT_OCCUPANCY

- **Title:** EOT Occupancy
- **Category:** RIX
- **[Bit(s)] Value:** [21]0x0, **Max. Inc/Cyc:** 1,
- **Definition:** Used with MC field. Report a rolling count whenever a 7b counter (count == 128) overflows for the selected MC's allocation into EOT.
- **NOTE:** This basically counts VNA.

Table 2-57. Unit Masks for EOT_ROLL_DEPTH_ACC

| Extension | IPERF Bit Values [20:17] | Description |
|-----------|--------------------------|------------------------------|
| --- | b0000 | (*nothing will be counted*) |
| HOM_VN0 | b0001 | Home (VN0) Messages |
| HOM_VN1 | b0010 | Home (VN1) Messages |
| NDR | b0011 | Non-Data Response Messages |
| DRS_VN0 | b0100 | Data Response (VN0) Messages |

Table 2-57. Unit Masks for EOT_ROLL_DEPTH_ACC

| Extension | IPERF Bit Values [20:17] | Description |
|-----------|--------------------------|--------------------------------|
| SNP | b0101 | Snoop Messages |
| NCB | b0110 | Non-Coherent Bypass Messages |
| NCS | b0111 | Non-Coherent Standard Messages |
| DRS_VN1 | b1000 | Data Response (VN1) Messages |
| --- | b1001-b1111 | (*nothing will be counted*) |

FLITS_RECV_ERR

- **Title:** Error Flits Received
- **Category:** RIX
- **[Bit(s)] Value:** [24]0x1, **Max. Inc/Cyc:** 1,
- **Definition:** Counts all flits received which caused CRC Error.

FLITS_RECV_SPEC

- **Title:** Special Flits Received
- **Category:** RIX
- **[Bit(s)] Value:** [25]0x1, **Max. Inc/Cyc:** 1,
- **Definition:** Counts all special flits received.

FLITS_SENT

- **Title:** Flits Sent
- **Category:** RIX
- **[Bit(s)] Value:** [31]0x1, **Max. Inc/Cyc:** 1,
- **Definition:** Counts all flits. Output queue receives active beat.

GLOBAL_ARB_BID

- **Title:** Global ARB Bids
- **Category:** QLX
- **[Bit(s)] Value:** [3:0]0x2, **Max. Inc/Cyc:** 1,
- **Definition:** Count global arbitration bids from the port.

GLOBAL_ARB_BID_FAIL

- **Title:** Failed Global ARB Bids
- **Category:** QLX
- **[Bit(s)] Value:** [3:0]0x5, **Max. Inc/Cyc:** 1,
- **Definition:** Number of bids for output port that were rejected at the global ARB.

Table 2-58. Unit Masks for GLOBAL_ARB_BID_FAIL

| Extension | QLX_EVENT_CFG Bit Values [7:4] | Description |
|-----------|--------------------------------|------------------------------------|
| VN0.HOM | b0000 | VN0 Home Messages |
| VN0.SNP | b0001 | VN0 Snoop Messages |
| VN0.NDR | b0010 | VN0 Non-Data Response Messages |
| VN0.NCS | b0011 | VN0 Non-Coherent Standard Messages |
| VN0.DRS | b0100 | VN0 Data Response Messages |
| VN0.NCB | b1001 | VN0 Non-Coherent Bypass Messages |
| --- | b0110-b0111 | (*illegal selection*) |
| VN1.HOM | b1000 | VN1 Home Messages |
| VN1.SNP | b1001 | VN1 Snoop Messages |
| VN1.NDR | b1010 | VN1 Non-Data Response Messages |
| VN1.NCS | b1011 | VN1 Non-Coherent Standard Messages |
| VN1.DRS | b1100 | VN1 Data Response Messages |
| VN1.NCB | b1101 | VN1 Non-Coherent Bypass Messages |
| --- | b1110-b1111 | (*illegal selection*) |

INQUE_READ_WIN

- **Title:** Input Queue Read Win.
- **Category:** RIX
- **[Bit(s)] Value:** [8]0x1, **Max. Inc/Cyc:** 1,
- **Definition:** Bid wins arbitration. Counts number of IQA reads and drains to XBAR.

LOCAL_ARB_BID

- **Title:** Local ARB Bids
- **Category:** QLX
- **[Bit(s)] Value:** [3:0]0x1, **Max. Inc/Cyc:** 1,
- **Definition:** Number of clocks of non-zero Local ARB Bids.

LOCAL_ARB_BID_FAIL

- **Title:** Failed Local ARB Bids
- **Category:** QLX
- **[Bit(s)] Value:** [3:0]0x4, **Max. Inc/Cyc:** 1,
- **Definition:** Number of clocks of non-zero Local ARB Bids that were rejected.

NEW_PACKETS_RECV

- **Title:** New Packets Received by Port
- **Category:** RIX
- **[Bit(s)] Value:** see table, **Max. Inc/Cyc:** 1,
- **Definition:** Counts new packets received according to the Virtual Network and Message Class specified.
- **NOTE:** Any combination of Message Class [5:0] may be monitored.

Table 2-59. Unit Masks for NEW_PACKETS_RECV

| Extension | IPERF Bit Values [7:0] | Description |
|-----------|------------------------|------------------------------------|
| VN0.NCB | b00xxxxx1 | VN0 Non-Coherent Bypass Messages |
| VN0.NCS | b00xxxx1x | VN0 Non-Coherent Standard Messages |
| VN0.DRS | b00xxx1xx | VN0 Data Response Messages |
| VN0.NDR | b00xx1xxx | VN0 Non-Data Response Messages |
| VN0.SNP | b00x1xxxx | VN0 Snoop Messages |
| VN0.HOM | b001xxxxx | VN0 Home Messages |
| VN0.ALL | b00111111 | VN0 All Messages |
| VN1.NCB | b01xxxxx1 | VN1 Non-Coherent Bypass Messages |
| VN1.NCS | b01xxxx1x | VN1 Non-Coherent Standard Messages |
| VN1.DRS | b01xxx1xx | VN1 Data Response Messages |
| VN1.NDR | b01xx1xxx | VN1 Non-Data Response Messages |
| VN1.SNP | b01x1xxxx | VN1 Snoop Messages |
| VN1.HOM | b011xxxxx | VN1 Home Messages |
| VN1.ALL | b01111111 | VN1 All Messages |
| VNA.NCB | b10xxxxx1 | VNA Non-Coherent Bypass Messages |
| VNA.NCS | b10xxxx1x | VNA Non-Coherent Standard Messages |
| VNA.DRS | b10xxx1xx | VNA Data Response Messages |
| VNA.NDR | b10xx1xxx | VNA Non-Data Response Messages |
| VNA.SNP | b10x1xxxx | VNA Snoop Messages |
| VNA.HOM | b101xxxxx | VNA Home Messages |
| VNA.ALL | b10111111 | VNA All Messages |
| ANY.NCB | b11xxxxx1 | ANY Non-Coherent Bypass Messages |
| ANY.NCS | b11xxxx1x | ANY Non-Coherent Standard Messages |
| ANY.DRS | b11xxx1xx | ANY Data Response Messages |
| ANY.NDR | b11xx1xxx | ANY Non-Data Response Messages |
| ANY.SNP | b11x1xxxx | ANY Snoop Messages |
| ANY.HOM | b111xxxxx | ANY Home Messages |
| ANY.ALL | b11111111 | ANY All Messages |

NULL_IDLE

- **Title:** Null Idle Flits
- **Category:** RIX
- **[Bit(s)] Value:** [30]0x1, **Max. Inc/Cyc:** 1,
- **Definition:** Counts all null idle flits sent.

OUTPUTQ_NE

- **Title:** Output Queue Not Empty
- **Category:** RIX
- **[Bit(s)] Value:** [26]0x1, **Max. Inc/Cyc:** 1,
- **Definition:** Output Queue Not Empty in this Output Port.

OUTPUTQ_OVFL

- **Title:** Output Queue Overflowed
- **Category:** RIX
- **[Bit(s)] Value:** [27]0x1, **Max. Inc/Cyc:** 1,
- **Definition:** Output Queue Overflowed in this Output Port.

QUE_ARB_BID

- **Title:** Queue ARB Bids
- **Category:** • **Category:** QLX
- **[Bit(s)] Value:** [3:0]0x0, **Max. Inc/Cyc:** 1,
- **Definition:** Number of Queue ARB Bids for specified messages (Number of clocks with non-zero bids from that queue).

Table 2-60. Unit Masks for QUE_ARB_BID

| Extension | QLX_EVENT_CFG Bit Values [6:4] | Description |
|-----------|--------------------------------|--------------------------------|
| HOM | b000 | Home Messages |
| SNP | b001 | Snoop Messages |
| NDR | b010 | Non-Data Response Messages |
| NCS | b011 | Non-Coherent Standard Messages |
| DRS | b100 | Data Response Messages |
| NCB | b101 | Non-Coherent Bypass Messages |
| --- | b110-b111 | (*illegal selection*) |

QUE_ARB_BID_FAIL

- **Title:** Failed Queue ARB Bids
- **Category:** QLX
- **[Bit(s)] Value:** [3:0]0x3, **Max. Inc/Cyc:** 1,
- **Definition:** Number of Queue ARB bids for input port that were rejected.

Table 2-61. Unit Masks for QUE_ARB_BID_FAIL

| Extension | QLX_EVENT_ CFG Bit Values [6:4] | Description |
|-----------|--|--------------------------------|
| HOM | b000 | Home Messages |
| SNP | b001 | Snoop Messages |
| NDR | b010 | Non-Data Response Messages |
| NCS | b011 | Non-Coherent Standard Messages |
| DRS | b100 | Data Response Messages |
| NCB | b101 | Non-Coherent Bypass Messages |
| --- | b110-b111 | (*illegal selection*) |

RETRYQ_NE

- **Title:** Retry Queue Not Empty
- **Category:** RIX
- **[Bit(s)] Value:** [28]0x1, **Max. Inc/Cyc:** 1,
- **Definition:** Retry Queue Not Empty in this Output Port.

RETRYQ_OV

- **Title:** Retry Queue Overflowed
- **Category:** RIX
- **[Bit(s)] Value:** [29]0x1, **Max. Inc/Cyc:** 1,
- **Definition:** Retry Queue Overflowed in this Output Port.

STARVING

- **Title:** Starvation Detected
- **Category:** QLX
- **[Bit(s)] Value:** [3:0]0xA, **Max. Inc/Cyc:** 1,
- **Definition:** Starvation detected

TARGET_AVAILABLE

- **Title:** Target Available
- **Category:** QLX
- **[Bit(s)] Value:** [3:0]0x9, **Max. Inc/Cyc:** 1,
- **Definition:** Number of times target was available at output port.

Table 2-62. Unit Masks for TARGET_AVAILABLE

| Extension | QLX_EVENT_CFG Bit Values [7:4] | Description |
|-----------|--------------------------------|-------------------------------------|
| VN0.HOM | b0000 | VN0 Home Messages |
| VN0.SNP | b0001 | VN0 Snoop Messages |
| VN0.NDR | b0010 | VN0 Non-Data Response Messages |
| VN0.NCS | b0011 | VN0 Non-Coherent Standard Messages |
| VN0.DRS | b0100 | VN0 Data Response Messages |
| VN0.NCB | b1001 | VN0 Non-Coherent Bypass Messages |
| VN0.VSM | b0110 | VN0 VNA-small (<= 3 flits) Messages |
| VN0.VLG | b0111 | VN0 VNA-large (9-11 flits) Messages |
| VN1.HOM | b1000 | VN1 Home Messages |
| VN1.SNP | b1001 | VN1 Snoop Messages |
| VN1.NDR | b1010 | VN1 Non-Data Response Messages |
| VN1.NCS | b1011 | VN1 Non-Coherent Standard Messages |
| VN1.DRS | b1100 | VN1 Data Response Messages |
| VN1.NCB | b1101 | VN1 Non-Coherent Bypass Messages |
| VN1.VSM | b1110 | VN1 VNA-small (<= 3 flits) Messages |
| VN1.VLG | b1111 | VN1 VNA-large (9-11 flits) Messages |

2.7 M-Box Performance Monitoring

2.7.1 Overview of the M-Box

The memory controller interfaces to the Intel® (WSM-EX) Scalable Memory Buffers and translates read and write commands into specific Intel® Scalable Memory Interconnect (Intel® SMI) operations. Intel SMI is based on the FB-DIMM architecture, but the Intel (WSM-EX) Scalable Memory Buffer is not an AMB2 device and has significant exceptions to the FB-DIMM2 architecture. The memory controller also provides a variety of RAS features, such as ECC, memory scrubbing, thermal throttling, mirroring, and DIMM sparing. Each socket has two independent memory controllers, and each memory controller has two Intel SMI channels that operate in lockstep.

2.7.2 Functional Overview

The memory controller is the interface between the home node controller (B-Box) and the Intel Scalable Memory Interconnect and basically translates read and write commands into specific memory commands and schedules them with respect to memory timing. The other main function of the memory controller is advanced ECC support. There are two memory controllers per socket, each controlling two Intel SMI channels in lockstep. Because of the data path affinity to the B-Box data path, each B-Box is paired with a memory controller, that is, B-Boxes and memory controllers come in pairs.

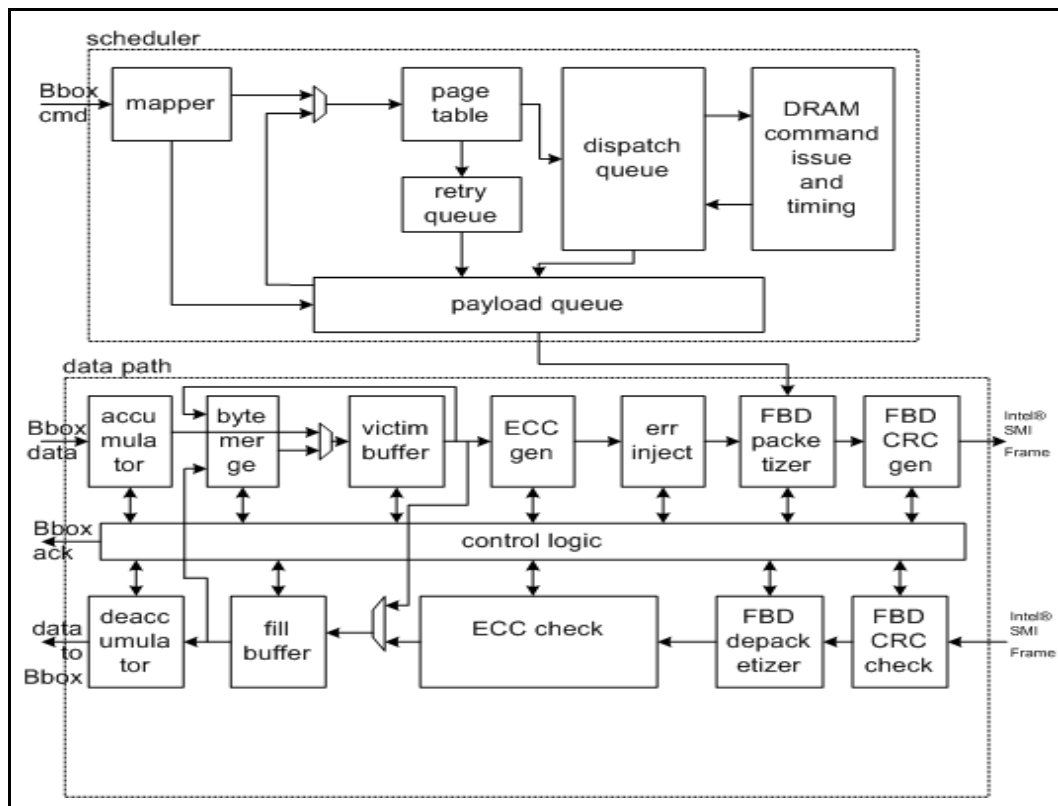


Figure 2-2. Memory Controller Block Diagram

The memory controller interfaces to the router through the B-Box (home node coherence controller) and to the P-Box pads.

2.7.2.1 Intel® Xeon® Processor E7 Family Scalable Memory Buffer

The Intel® Xeon® Processor E7 Family supports Scalable Memory Buffers on the Intel® SMI channels.

- Intel SMI protocol and signalling includes support for the following:
 - 4.8 Gbs, 6.4 Gbs signalling
 - forwarded clock fail-over NB and SB.
 - 9 data lanes plus 1 CRC lane plus 1 spare lane SB.
 - 12 data lanes plus 1 CRC lane plus 1 spare NB.
 - Support for integrating RDIMM thermal sensor information into Intel® SMI Status Frame.
- No support for daisy chaining (Intel (WSM-EX) Scalable Memory Buffer is the only Intel SMI device in the channel).
- No support for FB-DIMM1 protocol and signaling.

The Intel® Xeon® Processor E7 Family Scalable Memory Buffer provides an interface to DDR3 DIMMs and supports the following DDR3 functionality:

- DDR3 protocol and signalling, includes support for the following:
 - Up to two RDIMMs per DDR3 bus
 - Up to eight physical ranks per DDR3 bus (sixteen per Intel (WSM-EX) Scalable Memory Buffer)
 - 800 MT/s or 1066 MT/s (both DDR3 buses must operate at the same frequency)

- Single Rank x4, Dual Rank x4, Single Rank x8, Dual Rank x8, Quad Rank x4, Quad Rank x8
- 1 GB, 2 GB, 4 GB, 8 GB, 16 GB DIMM
- DRAM device sizes: 1 Gb, 2 Gb
- Mixed DIMM types (no requirement that DIMMs must be the same type, except that all DIMMs attached to an Intel® Xeon® Processor E7 Family Scalable Memory Buffer must run with a common frequency and core timings. Host lockstep requirements may impose additional requirements on DIMMs on separate Intel SMI channels).
- DDR buses may contain different number of DIMMs, zero through two. (Host lockstep requirements may impose additional requirements on DIMMs on separate Intel SMI channels).
- Cmd/Addr parity generation and error logging.
- No support for non-ECC DIMMs
- No support for DDR2 protocol and signaling
- Support for integrating RDIMM thermal sensor information into Intel SMI Status Frame.

2.7.3 M-Box Performance Monitoring Overview

Each M-Box supports performance monitoring through 6 48-bit wide counters (M_MSR_PMU_CNT_{5-0}). Each of these counters can be configured to monitor any available event through its companion control register. However, a good chunk of the generic events defer configuration to various subcontrol registers (as detailed below). Since there are a limited number of control registers, software must pay attention to various restrictions as to what events may be counted simultaneously. The M-Box counters increment by a maximum of 1 per cycle.

For information on how to setup a monitoring session, refer to [Section 2.1, “Global Performance Monitoring Control Counter Overflow”](#).

2.7.3.1 Choosing An Event To Monitor - Example using subcontrol registers

As has been stated, monitoring a particular event often requires configuring auxiliary MSRs.

For instance, to count (in counter 0) the number of RAS DRAM commands (PLD_DRAM_EV.DRAM_CMD.RAS) that have been issued, set up is as follows:

```
M_MSR_PMU_CNT_CTL_0.en [0] = 1
M_MSR_PMU_CNT_CTL_0.count_mode [3:2] = 0x0
M_MSR_PMU_CNT_CTL_0.flag_mode [7] = 0
M_MSR_PMU_CNT_CTL_0.inc_sel [13:9] = 0xa
M_MSR_PMU_PLD.cmd [0] = 0x0
M_MSR_PMU_PLD.dram_cmd [12:8] = 0x2
```

To count (in counter 2) the number of read commands from the B-Box in the scheduler queue (BCMD_SCHEDQ_OCCUPANCY.READS), set up is as follows:

```
M_MSR_PMU_CNT_CTL_2.en [0] = 1
M_MSR_PMU_CNT_CTL_0.count_mode [3:2] = 0x0
M_MSR_PMU_CNT_CTL_0.flag_mode [7] = 1
M_MSR_PMU_CNT_CTL_0.inc_sel [21:19] = 0x06
```

`M_MSR_PMU_MAP.cmd [8:5] = 0x0`

`M_MSR_PMU_MAP.fvid [3:0] = selected Fill Victim Index to track`

2.7.3.2 M-Box PMU - Overflow, Freeze and Unfreeze

If an overflow is detected from an M-Box performance counter, the overflow bit is set at the box level (`M_MSR_PMON_GLOBAL_STATUS.ov`), and forwarded up the chain towards the U-Box. If a M-Box0 counter overflows, a notification is sent and stored in S-Box0 (`S_MSR_PMON_SUMMARY.ov_mb`) which, in turn, sends the overflow notification up to the U-Box (`U_MSR_PMON_GLOBAL_STATUS.ov_s0`). Refer to [Table 2-26, “S_MSR_PMON_SUMMARY Register Fields”](#) to determine how each M-Box’s overflow bit is accumulated in the attached S-Box.

HW can be also configured (by setting the corresponding `.pmi_en` to 1) to send a PMI to the U-Box when an overflow is detected. The U-Box may be configured to freeze all uncore counting and/or send a PMI to selected cores when it receives this signal.

Once a freeze has occurred, in order to see a new freeze, the overflow field responsible for the freeze, must be cleared by setting the corresponding bit in `M_MSR_PMON_GLOBAL_OVF_CTL.clr_ov`. Assuming all the counters have been locally enabled (`.en` bit in data registers meant to monitor events) and the overflow bit(s) has been cleared, the M-Box is prepared for a new sample interval. Once the global controls have been re-enabled ([Section 2.1.5, “Enabling a New Sample Interval from Frozen Counters”](#)), counting will resume.

2.7.4 M-BOX Performance Monitors

Table 2-63. M-Box Performance Monitoring MSRs

| MSR Name | Access | MSR Address | Size (bits) | Description |
|----------------------------------|--------|-------------|-------------|--|
| MB1_CR_M_MSR_PMU_ADDR_MASK | RW_RW | 0x0E5E | 64 | M-Box 1 PMON Addr Mask |
| MB1_CR_M_MSR_PMU_ADDR_MATCH | RW_RW | 0x0E5D | 64 | M-Box 1 PMON Addr Match |
| MB1_CR_M_MSR_PMU_MM_CFG | RW_RW | 0xCE5C | 64 | M-Box 1 PMON Mask/Match Config |
| | | | | |
| MBO_CR_M_MSR_PMU_ADDR_MASK | RW_RW | 0x0E56 | 64 | M-Box 0 PMON Addr Mask |
| MBO_CR_M_MSR_PMU_ADDR_MATCH | RW_RW | 0x0E55 | 64 | M-Box 0 PMON Addr Match |
| MBO_CR_M_MSR_PMU_MM_CFG | RW_RW | 0x0E54 | 64 | M-Box 0 PMON Mask/Match Config |
| | | | | |
| MB1_CR_M_MSR_PMU_CNT_5 | RW_RW | 0x0CFB | 64 | M-Box 1 PMON Counter 5 |
| MB1_CR_M_MSR_PMU_CNT_CTL_5 | RW_RW | 0x0CFA | 64 | M-Box 1 PMON Control 5 |
| MB1_CR_M_MSR_PMU_CNT_4 | RW_RW | 0x0CF9 | 64 | M-Box 1 PMON Counter 4 |
| MB1_CR_M_MSR_PMU_CNT_CTL_4 | RW_RW | 0x0CF8 | 64 | M-Box 1 PMON Control 4 |
| MB1_CR_M_MSR_PMU_CNT_3 | RW_RW | 0x0CF7 | 64 | M-Box 1 PMON Counter 3 |
| MB1_CR_M_MSR_PMU_CNT_CTL_3 | RW_RW | 0x0CF6 | 64 | M-Box 1 PMON Control 3 |
| MB1_CR_M_MSR_PMU_CNT_2 | RW_RW | 0x0CF5 | 64 | M-Box 1 PMON Counter 2 |
| MB1_CR_M_MSR_PMU_CNT_CTL_2 | RW_RW | 0x0CF4 | 64 | M-Box 1 PMON Control 2 |
| MB1_CR_M_MSR_PMU_CNT_1 | RW_RW | 0x0CF3 | 64 | M-Box 1 PMON Counter 1 |
| MB1_CR_M_MSR_PMU_CNT_CTL_1 | RW_RW | 0x0CF2 | 64 | M-Box 1 PMON Control 1 |
| MB1_CR_M_MSR_PMU_CNT_0 | RW_RW | 0x0CF1 | 64 | M-Box 1 PMON Counter 0 |
| MB1_CR_M_MSR_PMU_CNT_CTL_0 | RW_RW | 0x0CF0 | 64 | M-Box 1 PMON Control 0 |
| | | | | |
| MB1_CR_M_MSR_PMU_ZDP_CTL_FVC | RW_RW | 0x0CEB | 32 | M-Box 1 PMON SubControl for FVC events |
| MB1_CR_M_MSR_PMU_PLD | RW_RW | 0x0CEA | 32 | M-Box 1 PMON SubControl for PLD events |
| MB1_CR_M_MSR_PMU_PGT | RW_RW | 0x0CE9 | 32 | M-Box 1 PMON SubControl for PGT events |
| MB1_CR_M_MSR_PMU_MSC_THR | RW_RW | 0x0CE8 | 32 | M-Box 1 PMON SubControl for THR events |
| MB1_CR_M_MSR_PMU_MAP | RW_RW | 0x0CE7 | 32 | M-Box 1 PMON SubControl for MAP events |
| MB1_CR_M_MSR_PMU_ISS | RW_RW | 0x0CE6 | 32 | M-Box 1 PMON SubControl for ISS events |
| MB1_CR_M_MSR_PMU_DSP | RW_RW | 0x0CE5 | 32 | M-Box 1 PMON SubControl for DSP events |
| MB1_CR_M_MSR_PMU_TIMESTAMP_UNIT | RW_RW | 0x0CE4 | 32 | M-Box 1 PMON Timestamp |
| | | | | |
| MB1_CR_M_MSR_PERF_GLOBAL_OVF_CTL | RW_RW | 0x0CE2 | 32 | M-Box 1 PMON Global Overflow Control |
| MB1_CR_M_MSR_PERF_GLOBAL_STATUS | RW_RW | 0x0CE1 | 32 | M-Box 1 PMON Global Overflow Status |
| MB1_CR_M_MSR_PERF_GLOBAL_CTL | RW_RW | 0x0CE0 | 32 | M-Box 1 PMON Global Control |

| MSR Name | Access | MSR Address | Size (bits) | Description |
|----------------------------------|--------|-------------|-------------|--|
| MBO_CR_M_MSR_PMU_CNT_5 | RW_RW | 0x0CBB | 64 | M-Box 0 PMON Counter 5 |
| MBO_CR_M_MSR_PMU_CNT_CTL_5 | RW_RW | 0x0CBA | 64 | M-Box 0 PMON Control 5 |
| MBO_CR_M_MSR_PMU_CNT_4 | RW_RW | 0x0CB9 | 64 | M-Box 0 PMON Counter 4 |
| MBO_CR_M_MSR_PMU_CNT_CTL_4 | RW_RW | 0x0CB8 | 64 | M-Box 0 PMON Control 4 |
| MBO_CR_M_MSR_PMU_CNT_3 | RW_RW | 0x0CB7 | 64 | M-Box 0 PMON Counter 3 |
| MBO_CR_M_MSR_PMU_CNT_CTL_3 | RW_RW | 0x0CB6 | 64 | M-Box 0 PMON Control 3 |
| MBO_CR_M_MSR_PMU_CNT_2 | RW_RW | 0x0CB5 | 64 | M-Box 0 PMON Counter 2 |
| MBO_CR_M_MSR_PMU_CNT_CTL_2 | RW_RW | 0x0CB4 | 64 | M-Box 0 PMON Control 2 |
| MBO_CR_M_MSR_PMU_CNT_1 | RW_RW | 0x0CB3 | 64 | M-Box 0 PMON Counter 1 |
| MBO_CR_M_MSR_PMU_CNT_CTL_1 | RW_RW | 0x0CB2 | 64 | M-Box 0 PMON Control 1 |
| MBO_CR_M_MSR_PMU_CNT_0 | RW_RW | 0x0CB1 | 64 | M-Box 0 PMON Counter 0 |
| MBO_CR_M_MSR_PMU_CNT_CTL_0 | RW_RW | 0x0CB0 | 64 | M-Box 0 PMON Control 0 |
| MBO_CR_M_MSR_PMU_ZDP_CTL_FVC | RW_RW | 0x0CAB | 32 | M-Box 0 PMON SubControl for FVC events |
| MBO_CR_M_MSR_PMU_PLD | RW_RW | 0x0CAA | 32 | M-Box 0 PMON SubControl for PLD events |
| MBO_CR_M_MSR_PMU_PGT | RW_RW | 0x0CA9 | 32 | M-Box 0 PMON SubControl for PGT events |
| MBO_CR_M_MSR_PMU_MSC_THR | RW_RW | 0x0CA8 | 32 | M-Box 0 PMON SubControl for THR events |
| MBO_CR_M_MSR_PMU_MAP | RW_RW | 0x0CA7 | 32 | M-Box 0 PMON SubControl for MAP events |
| MBO_CR_M_MSR_PMU_ISS | RW_RW | 0x0CA6 | 32 | M-Box 0 PMON SubControl for ISS events |
| MBO_CR_M_MSR_PMU_DSP | RW_RW | 0x0CA5 | 32 | M-Box 0 PMON SubControl for DSP events |
| MBO_CR_M_MSR_PMU_TIMESTAMP_UNIT | RW_RW | 0x0CA4 | 32 | M-Box 0 PMON Timestamp |
| MBO_CR_M_MSR_PERF_GLOBAL_OVF_CTL | RW_RW | 0x0CA2 | 32 | M-Box 0 PMON Global Overflow Control |
| MBO_CR_M_MSR_PERF_GLOBAL_STATUS | RW_RW | 0x0CA1 | 32 | M-Box 0 PMON Global Overflow Status |
| MBO_CR_M_MSR_PERF_GLOBAL_CTL | RW_RW | 0x0CA0 | 32 | M-Box 0 PMON Global Control |

2.7.4.1 M-Box Box Level PMON state

The following registers represent the state governing all box-level PMUs in the M-Box.

The `_GLOBAL_CTL` register contains the bits used to enable monitoring. It is necessary to set the `.ctr_en` bit to 1 before the corresponding data register can collect events.

If an overflow is detected from one of the M-Box PMON registers, the corresponding bit in the `_GLOBAL_STATUS.ov` field will be set. To reset the overflow bits set in the `_GLOBAL_STATUS.ov` field, a user must set the corresponding bits in the `_GLOBAL_OVF_CTL.clr_ov` field before beginning a new sample interval.

Table 2-64. M_MSR_PERF_GLOBAL_CTL Register Fields

| Field | Bits | HW Reset Value | Description |
|--------|------|----------------|--|
| ctr_en | 5:0 | 0 | Must be set to enable each MBOX 0 counter (bit 0 to enable ctr0, etc) NOTE: U-Box enable and per counter enable must also be set to fully enable the counter. |

Table 2-65. M_MSR_PERF_GLOBAL_STATUS Register Fields

| Field | Bits | HW Reset Value | Description |
|-------|------|----------------|--|
| ov | 5:0 | 0 | If an overflow is detected from the corresponding MBOX 0 PMON register, it's overflow bit will be set. |

Table 2-66. M_MSR_PERF_GLOBAL_OVF_CTL Register Fields

| Field | Bits | HW Reset Value | Description |
|--------|------|----------------|--|
| clr_ov | 5:0 | 0 | Writing '1' to bit in field causes corresponding bit in 'Overflow PerfMon Counter' field in MBO_CR_M_MSR_PERF_GLOBAL_STATUS register to be cleared to 0. |

2.7.4.2 M-Box PMON state - Counter/Control Pairs

The following table defines the layout of the M-Box performance monitor control registers. The main task of these configuration registers is to select the event to be monitored by their respective data counter. Setting the *.inc_sel* and *.set_flag_sel* fields performs the event selection (alternatively, a user can use *.dec_sel* and *.rst_flag_sel* to use a decrementing form of the event select). Many of the events selected may be broken into components through use of companion subcontrol registers. See [Section 2.7.7, "M-Box Performance Monitor Event List"](#) for more details.

The *.en* bit must be set to 1 to enable counting.

Additional control bits include:

- *.pmi_en* governs what to do if an overflow is detected.

Table 2-67. M_MSR_PMU_CNT_CTL{5-0} Register – Field Definitions

| Field | Bits | HW Reset Value | Description |
|--------------|-------|----------------|--|
| ig | 63 | 0 | Read zero; writes ignored. |
| rsv | 62:61 | 0 | Reserved; Must write to 0 else behavior is undefined. |
| ig | 60:25 | 0 | Read zero; writes ignored. |
| rsv | 24:22 | 0 | Reserved; Must write to 0 else behavior is undefined. |
| set_flag_sel | 21:19 | 0 | Selects the 'set' condition for enable flag. Secondary event select. See Table 2-84, "Performance Monitor Events for M-Box Events" for events elected by this field. NOTE: Bit 7 (flag_mode) must be set to 1 to enable this field. |
| rsv | 18:14 | 0 | Reserved; Must write to 0 else behavior is undefined. |
| inc_sel | 13:9 | 0 | Selects increment signal for this counter. Primary event select. See Table 2-84, "Performance Monitor Events for M-Box Events" for events elected by this field. |
| rsv | 8 | 0 | Reserved; Must write to 0 else behavior is undefined. |
| flag_mode | 7 | 0 | Enable conditional counting using set_flag_sel |
| wrap_mode | 6 | 0 | Counter wrap mode. If set to 0, this counter will stop counting on detection of over/underflow. If set to 1, this counter will wrap and continue counting on detection of over/underflow. |
| storage_mode | 5:4 | 0 | Storage mode. If set to 0, no count enable flag is required. If set to 1, count enable flag must have a value of 1 for counting to occur. |
| count_mode | 3:2 | 0 | 00 - count will increase (up) 01 - count will decrease (dn) 10 - count can increase and decrease |
| pmi_en | 1 | 0 | Enable PMON interrupt on counter over/underflow. |
| en | 0 | 0 | Local Counter Enable. When set, the associated counter is locally enabled. NOTE: It must also be enabled in M_MSR_PMON_GLOBAL_CTL and the U-Box to be fully enabled. |

The M-Box performance monitor data registers are 48b wide. A counter overflow occurs when a carry out bit from bit 47 is detected. Software can force all uncore counting to freeze after N events by preloading a monitor with a count value of $(2^{48} - 1) - N$ and setting the control register to send a PMI to the U-Box. Upon receipt of the PMI, the U-Box will disable counting ([Section 2.1.1, "Freezing on Counter Overflow"](#)). During the interval of time between overflow and global disable, the counter value will wrap and continue to collect events.

In this way, software can capture the precise number of events that occurred between the time uncore counting was enabled and when it was disabled (or 'frozen') with minimal skew.

If accessible, software can continuously read the data registers without disabling event collection.

Table 2-68. M_MSR_PMU_CNT_{5-0} Register – Field Definitions

| Field | Bits | HW Reset Value | Description |
|-------------|------|----------------|----------------------------------|
| event_count | 47:0 | 0 | 48-bit performance event counter |

The M-Box also includes a 16b timestamp unit that is incremented each M-Box clock tick. It is a free-running counter unattached to the rest of the M-Box PMU, meaning it does not generate an event fed to the other counters.

Table 2-69. M_MSR_PMU_TIMESTAMP_UNIT Register – Field Definitions

| Field | Bits | HW Reset Value | Description |
|-----------|------|----------------|--|
| timestamp | 15:0 | 0 | Timestamp is updated every timestamp_unit MClk's |

2.7.4.3 M-Box PMU Filter Registers

The M-Box also provides a limited ability to perform address matching for PLD events. The following 3 tables contain the field definitions for the configuration registers governing the M-Box's address match/mask facility.

Table 2-70. M_MSR_PMU_MM_CFG Register – Field Definitions

| Field | Bits | HW Reset Value | Description |
|--------|------|----------------|---|
| enable | 63 | 0 | Enable debug mode (disables PMON mode). |
| ig | 62:0 | 0 | Read zero; writes ignored. |

Table 2-71. M_MSR_PMU_ADDR_MATCH Register – Field Definitions

| Field | Bits | HW Reset Value | Description |
|---------|-------|----------------|----------------------------------|
| ig | 63:34 | 0 | Read zero; writes ignored. |
| address | 33:0 | 0 | 34b match address for PLD events |

Table 2-72. M_MSR_PMU_ADDR_MASK Register – Field Definitions

| Field | Bits | HW Reset Value | Description |
|---------|-------|----------------|--|
| ig | 63:28 | 0 | Read zero; writes ignored. |
| address | 27:0 | 0 | Address bits to mask 'don't care' bits during match - cache aligned address 33:6 |

2.7.4.4 M-Box PMU Subcontrol Registers - Subunit descriptions

The following Tables contain information on how to program the various subcontrol registers contained within the M-Box which include the DSP, ISS, MAP, THR, PGT, PLD and FVC registers. The subcontrol registers govern events coming from subunits within the M-Box which can be roughly categorized as follows:

MAP - Memory Mapper - receives read and write commands/addresses from the B-Box and translates the received addresses (physical) into DRAM addresses (rank, bank, row and column). The commands and translated addresses are sent to the PLD. In parallel, the broken DRAM addresses are also sent to the PGT.

PLD - Payload Queue - Receives command and translated addresses from the MAP while the PGT translates MAP commands into DRAM command combinations.

Original B-Box transaction's FVID sent from DSP during subcommand execution where the appropriate subcommand information is accessed to compose the Intel® SMI command frame.

PGT - Page Table - Keeps track of open pages. Translates the read/write commands into DRAM command combinations (i.e. PRE, RAS, CASrd, CASwr). The generated command combination (e.g. PRE_RAS_CASrd) is then sent to the Dispatch Queue.

If

a) there is already a command in the DSP for a particular DIMM (rank/bank)

b) the DSP's readQ or writeQ is full

c) if a rank requires thermal throttling because the DIMM is heating up

d) or a refresh is executing to a rank.

Then the PGT will detect the conflict and place the command in the retryQ for later execution.

DSP - Dispatch Queue - receives DRAM command from PGT and stores request in a read or write subqueue. In the dispatch queue, the command combinations are broken up into subcommand kinds that are sequenced in the necessary order required to complete the read/write transaction (i.e. PRE, RAS, CAS, CASpre). All "ready to execute" subcommands stored within the various DSP queues are presented simultaneously to the issue logic.

Once the ISS returns the subcommand choice, the oldest DSP entry containing that subcommand kind (for a particular DIMM) is allowed to execute. During subcommand execution, the DSP sends the original (B-Box) transaction's FVID (that was stored in the DSP entry) to the PLD. After subcommand execution, the DSP's queue entry state is updated to the next required subcommand kind (based on the original command combination) to be executed (new state).

ISS - Issue - receives "ready to execute" subcommands from the dispatch queue(s) as a bit vector that is organized with each bit representing a subcommand kind per DIMM (i.e. RAS for DIMM0, CAS for DIMM3). Having an overview of all these subcommand kinds enables the ISS to flexibly schedule/combine subcommands out-of-order. Once a subcommand kind for a particular DIMM is selected from the issue vector by the ISS, that subcommand choice is driven back to the DSP

THR - Thermal Throttling

FVC - Fill and Victim Control - drives all the control signals required by the fill datapath and victim datapath. Additionally, it handles issuing and control of the buffer maintenance commands (i.e. MRG, F2V, V2V, V2F and F2B). It also contains the logic to respond to the B-Box when commands in the M-Box have completed.

The **DSP** subcontrol register contains bits to specify subevents of the DSP_FILL event, breaking it into write queue/read queue occupancy as well as DSP latency.

Table 2-73. M_MSR_PMU_DSP Register – Field Definitions

| Field | Bits | HW Reset Value | Description |
|------------|-------|----------------|--|
| ig | 63:11 | 0 | Read zero; writes ignored. |
| wrq_empty | 10 | 0 | Generate DSP_FILL trigger when write queue is empty |
| rdq_empty | 9 | 0 | Generate DSP_FILL trigger when read queue is empty |
| wrq_full | 8 | 0 | Generate DSP_FILL trigger when write queue is full |
| rdq_full | 7 | 0 | Generate DSP_FILL trigger when read queue is full |
| lat_cnt_en | 6 | 0 | Latency count mode. If 1, the latency for this FVID is counted. |
| fvid | 5:0 | 0 | FVID (Fill Victim Index) of transaction for which scheduler latency is to be counted. Only fully completed transactions are counted. |

The **ISS** subcontrol register contains bits to specify subevents for the ISS_EV (by Intel SMI frame), CYCLES_SCHED_MODE (cycles spent per ISS mode) and PLD_DRAM_EV (DRAM commands broken down by scheduling mode in the ISS) events.

Table 2-74. M_CSR_ISS_PMU Register – Field Definitions

| Field | Bits | Access | HW Reset Value | Reset Type |
|---------------------|-------|--------|----------------|---|
| ig | 31:11 | | | Reads 0; writes ignored. |
| frame_beat | 10 | RW | 0 | Number of Frame Beats |
| sched_mode_pld_trig | 9:7 | RW | 0 | Selects the scheduling mode for which the number of DRAM commands is counted in MA_PLD. Here for implementation reasons. Uses same encodings as M_MSR_PMU_ISS.sched_mode: 000: trade-off 001: rd priority 010: wr priority 011: adaptive |
| sched_mode | 6:4 | RW | 0 | Selects the scheduling mode for which time-in-mode is counted. 000: trade-off 001: rd priority 010: wr priority 011: adaptive |
| frm_type | 3:0 | RW | 0 | Selects the frame type to be counted. 0000 - 3CMD - Count all 3-command Intel SMI frames 0001 - WDAT - Count all write data frames. 0010 - SYNC - Count all SYNC frames. 0011 - CHNL - Count all channel command frames. 0101 - 0100 - RSVD 1000 - NOP - Count all NOP frames. For post-silicon debug 1001-1011 - RSVD 1100 - Count all 1-command Intel SMI frames. 1101-1111 - RSVD |

The **MAP** subcontrol register contains bits to specify subevents for BCMD_SCHEDQ_OCCUPANCY (by B-Box command type).

Table 2-75. M_MSR_PMU_MAP Register – Field Definitions

| Field | Bits | HW Reset Value | Reset Type |
|----------------|-------|----------------|---|
| ig | 31:13 | | Reads 0; writes ignored. |
| set_patrol_req | 12:11 | 0 | Select various patrol requests for MAP PMU Event 2. 11 - Count all patrol requests accepted by PGT. 10 - Count write requests by Patrol FSM. The write does not need to be accepted by PGT. 01 - Count read requests by Patrol FSM. Read does not need to be accepted by PGT. 00 - Count every patrol request accepted by PGT |
| sel_map_ev3 | 10 | 0 | Selects MAP PMU Event 3. 1 - Count each time a trigger set up in MAP fires. 0 - Count every valid response from MBOX. This includes responses to B-Box as well as response to patrol scrub FSM. |
| anycmd | 9 | 0 | Count all B-Box commands to M-Box. Event is counted by PGT Event0. |
| cmd | 8:6 | 0 | B-Box command to count. Event is counted by PGT Event0. NOTE: anycmd MUST be 0. |
| fvid | 5:0 | 0 | Selects FVID (Fill Victim Index) for which overall or scheduler latency is to be counted. |

The **THR** subcontrol register contains bits to specify subevents for the THR_TT_TRP_UP/DN_EV events allowing a user to choose select DIMMs and whether the temperature is rising or falling.

Table 2-76. M_CSR_PMU_MA_MSC_THR Register – Field Definitions

| Field | Bits | Access | HW Reset Value | Reset Type |
|------------------|-------|--------|----------------|--|
| ig | 31:11 | | | Reads 0; writes ignored. |
| trp_pt_dn_cnd | 10:9 | RW | 0 | Selects the condition to count for “downwards” trip point crossings. See Table 2-77, “TRP_PT_{DN,UP}_CND Encodings” for encodings. |
| trp_pt_up_cnd | 8:7 | RW | 0 | Selects the condition to count for “upwards” trip point crossings. See Table 2-77, “TRP_PT_{DN,UP}_CND Encodings” for encodings. |
| dimms_trp_pt | 6:4 | RW | 0 | Selects the DIMM for which to count the trip point crossings. Unused when all_dimms_trp_pt field is set. |
| all_dimms_trp_pt | 3 | RW | 0 | Select all DIMMs to provide trip point crossings events instead of a single particular DIMM. |
| dimms_sel | 2:0 | RW | 0 | Selects DIMM to count the time throttling is active. |

Table 2-77. TRP_PT_{DN,UP}_CND Encodings

| Name | Value | Description |
|---------------------|-------|---|
| ABOVE_TEMP_MID_RISE | 0b11 | Above the mid temperature trip point (rising) |
| ABOVE_TEMP_MID_FALL | 0b10 | Above the mid temperature trip point (falling) |
| ABOVE_TEMP_LO | 0b01 | Above the low temperature trip point, but below the mid temperature trip point. |
| BELOW_TEMP_LO | 0b00 | Below the low temperature trip point. |

The **PGT** subcontrol register contains bits to specify subevents for CYCLES_PGT_STATE (time spent in open or closed state) and PGT_PAGE_EV (opn2cls or cls2opn transitions) as well as provide bits to further breakdown throttling events into ranks.

Table 2-78. M_MSR_PMU_PGT Register – Field Definitions

| Field | Bits | HW Reset Value | Reset Type |
|----------------|-------|----------------|--|
| ig | 31:12 | | Reads 0; writes ignored. |
| empty_or_clspg | 11 | 0 | Selects between counting number of page auto close commands or counting empty page hits. 0 - Auto Page Close Commands 1 - Empty Pages Hits by new access to page table |
| trig_sel | 10:9 | 0 | Selects PGT event to be counted. 00 - Time which PGT stays in open/closed page policy 01 - Number of PGT Trigger 1 matches 10 - Number of PGT Trigger 2 matches |
| trans_cmd_cnt | 8:7 | 0 | Selects translated commands to be counted. 00 - ALL - Count all translated commands. 01 - WR - Count all write commands. 10 - RD - Count all read commands. |
| opncls_time | 6 | 0 | Selects time counting between open and closed page mode. 0 - CLS - Counts time spent in closed page mode. 1 - OPN - Counts time spent in open page mode. |
| tt_rnk_cnd | 5:2 | 0 | Selects which rank is observed for thermal throttling events. |
| rnk_cnd | 1 | 0 | Selects how thermal throttling events are counted relative to rank. 0 - ALL - Counts thermal throttling events for all ranks. 1 - SGL - Counts thermal throttling events for the single rank selected by tt_rnk_cnd. |
| opn2cls_cnt | 0 | 0 | Counts the open/closed page policy transitions. 0 - OPN2CLS - Counts open-to-closed transactions. 1 - CLS2OPN - Counts closed-to-open transactions. |

The **PLD** subcontrol register contains bits to specify subevents for PLD_DRAM_EV (by DRAM CMD type), PLD_RETRY_EV (to specify FVID).

Table 2-79. M_MSR_PMU_PLD Register – Field Definitions

| Field | Bits | HW Reset Value | Reset Type |
|----------------------|-------|----------------|---|
| ig | 31:24 | | Reads 0; writes ignored. |
| dram_cmd1 | 23:19 | 0 | See dram_cmd field for enumeration |
| non_nop_trkl_cmd_cnt | 18 | 0 | NON NOP Trickle Commands |
| vld_cmd_cnt | 17 | 0 | Valid Commands |
| dram_cmd1_cnt | 16 | 0 | Count number of DRAM Commands scheduled (DRAM commands can be specified by setting the dram_cmd1 field to non-0) |
| pld_trig_sel | 15:14 | 0 | When 0, corresponding PMU event records number of ZAD parity errors. When 1 or 2, respective trigger match event is selected. |
| addr_match1 | 13 | 0 | Qualify trigger with address match as specified by M_CSR_INJ_ERR_ADDR_1, M_CSR_INJ_ERR_CTL_1.match_* and M_CSR_INJ_ERR_CTL_1.inj_err_* fields control the match condition. |
| dram_cmd | 12:8 | 0 | The DRAM command type to be counted. 11110 - ZQCAL_SCMD 11101 - RCR_SCMD 11100 - WCR_SCMD 11000 - NOWPE_SCMD 10111 - SFT_RST_SCMD 10110 - IBD_SCMD 10101 - CKEL_SCMD 10100 - CKEH_SCMD 10011 - POLL_SCMD 10010 - SYNC_SCMD 10001 - PRE_SCMD 10000 - TRKL_SCMD 01111 - GENDRM_SCMD 01110 - EMRS3_SCMD 01101 - EMRS2_SCMD 01100 - NOP_SCMD 01011 - EXSR_SCMD 01010 - ENSR_SCMD 01001 - RFR_SCMD 01000 - EMRS_SCMD 00111 - MRS_SCMD 00110 - CASPRE_WR_SCMD 00101 - CASPRE_RD_SCMD 00100 - CAS_WR_SCMD 00011 - (* undefined count *) 00010 - RAS_SCMD 00001 - PRECALL_SCMD 00000 - ILLEGAL_SCMD |
| rtry_sngl_fvid | 7 | 0 | Controls FVID (Fill Victim Index) selection for which the number of retries is to be counted. 0 - ALL - All retries are counted, regardless of FVID 1 - FVID - Counts only the retries whose FVIDs match this CSR's fvid field. |
| fvid | 6:1 | 0 | The FVID for which the number of retries is to be counted. |
| cmd | 0 | 0 | Uses setting in M_MSR_PMU_ISS.sched_mode to qualify DRAM commands. |

The **FVC** subcontrol register contains bits to break the FVC_EV into events observed by the Fill and Victim Control logic (i.e. B-Box commands, B-Box responses, various error conditions, etc). The FVC register can be set up to monitor four independent FVC-subevents simultaneously. However, many of the FVC-subevents depend on **additional** FVC fields which detail B-Box response and commands. Therefore, only one B-Box response or command may be monitored at any one time.

Table 2-80. M_MSR_PMU_ZDP_CTL_FVC Register – Field Definitions

| Field | Bits | HW Reset Value | Reset Type |
|---------------|-------|----------------|---|
| ig | 31:25 | | Reads 0; writes ignored. |
| pbox_init_err | 24 | 0 | Extension of event select encoding 0b111 (smi_nb_trig). If event select is set to 111 and this bit is set to 1, PBOX reset time events will be counted. |
| evnt3 | 23:21 | 0 | FVC Subevent 3 selection. See Table 2-81, "M_MSR_PMU_ZDP_CTL_FVC.evnt{4-1} Encodings" |
| evnt2 | 20:18 | 0 | FVC Subevent 2 selection. See Table 2-81, "M_MSR_PMU_ZDP_CTL_FVC.evnt{4-1} Encodings" |
| evnt1 | 17:15 | 0 | FVC Subevent 1 selection. See Table 2-81, "M_MSR_PMU_ZDP_CTL_FVC.evnt{4-1} Encodings" |
| evnt0 | 14:12 | 0 | FVC Subevent 0 selection. See Table 2-81, "M_MSR_PMU_ZDP_CTL_FVC.evnt{4-1} Encodings" |
| resp | 11:9 | 0 | B-Box response to match on. See Table 2-82, "M_MSR_PMU_ZDP_CTL_FVC.RESP Encodings" |
| bcmd | 8:6 | 0 | B-Box command to match on. See Table 2-82, "M_MSR_PMU_ZDP_CTL_FVC.RESP Encodings" |
| fvid | 5:0 | 0 | FVID to match on |

Table 2-81. M_MSR_PMU_ZDP_CTL_FVC.evnt{4-1} Encodings

| Name | Value | Description |
|-------------|-------|--|
| smi_nb_trig | 0b111 | Select Intel SMI Northbound debug event bits from the Intel SMI status frames as returned from the Intel (WSM-EX) Scalable Memory Buffers OR PBOX init error (see pbox_init_err field). These bits are denoted NBDE in the Intel SMI spec status frame description. An OR of all the bits over all the Intel (WSM-EX) Scalable Memory Buffers is selected here as an event. NB debug events generate multiple triggers for single NBDE event. Instead, the following triggers listed in M_CCSR_MSC_TRIG_SEL reg must be used: Trigger 51 := Ch1 NBDE Trigger 58 := Ch0 NBDE |
| resp_match | 0b110 | Use response match as programmed by Z_CSR_PMU_ZDP_CTL_FVC.resp to generate trigger. |
| bcmd_match | 0b101 | Use B-Box command match as programmed by Z_CSR_PMU_ZDP_CTL_FVC.bcmd to generate trigger. |
| fast_rst | 0b100 | Fast reset request from MBOS |
| alrt_frm | 0b011 | An alert frame was detected. |
| psn_txn | 0b010 | A write to memory was poisoned. |
| mem_ecc_err | 0b001 | Memory ECC error detected (that is not a link-level CRC error). |
| smi_crc_err | 0b000 | Link level Intel SMI CRC error detected. |

Table 2-82. M_MSR_PMU_ZDP_CTL_FVC.RESP Encodings

| Name | Value | Description |
|----------------|-------|---|
| spr_uncor_resp | 0b111 | Uncorrectable response for command to misbehaving DIMM during sparing. |
| Reserved | 0b110 | |
| spr_ack_resp | 0b101 | Positive acknowledgment for command to misbehaving DIMM during sparing. No error was detected for the transaction. |
| spec_ack_resp | 0b100 | Speculative (=early) positive acknowledgment for optimized read flow. No error was detected for the transaction. |
| uncor_resp | 0b011 | Uncorrectable response. Corrections failed. |
| corr_resp | 0b010 | Corrected (after, for example, error trials or just by a retry). |
| retry_resp | 0b001 | Retry response. Possibly a correctable error. Retries are generated until it is decided that error was either correctable or uncorrectable. |
| ack_resp | 0b000 | Positive acknowledgment. No error was detected. |

Table 2-83. M_MSR_PMU_ZDP_CTL_FVC.BCMD Encodings

| Name | Value | Description |
|------------|-------|--|
| sprwr_bcmd | 0b111 | Spare write. |
| f2b_bcmd | 0b110 | Fill buffer read to B-Box. |
| f2v_bcmd | 0b101 | Fill buffer to victim buffer transfer. |
| v2v_bcmd | 0b100 | Victim buffer to victim buffer transfer. |
| v2f_bcmd | 0b011 | Victim buffer to fill buffer transfer. |
| mrg_bcmd | 0b010 | Merge command from B-Box. |
| wr_bcmd | 0b001 | Memory write command from B-Box. |
| rd_bcmd | 0b000 | Memory read command from B-Box. |

2.7.5 M-Box Performance Monitoring Events

2.7.5.1 An Overview:

The M-box performance monitors can collect events in many of the substructures found within the M-Box including the DSP, ISS, MAP, THR, PGT, PLD and FVC (refer to [Section 2.7.4.4, "M-Box PMU Subcontrol Registers - Subunit descriptions"](#) for more detail).

A sampling of events available for monitoring in the M-Box:

- **B-Box commands** - reads, writes, fill2victim, merge, etc. Can be conditioned on fvid which allows determining average latency of M-Box and memory.
 - **B-Box responses**. Incrementing on read command and decrementing on read response allows one to determine the number of simultaneous reads in the M-Box. A max detector can log the max number of reads the M-Box received.
 - **Translated commands**: ras_caspre, ras_cas, cas, ras_cas_pre, pre, etc (can be filtered on r/w)
 - **Memory commands**: ras, cas, pre, prefetch, preall, etc.
 - **Page hits and page misses**.
 - Auto page closes.
 - Open-page to closed-**page policy transitions**. As well as length of time spent in each policy.
 - Starvation event in scheduler, starvation state and back-pressure to B-Box.
 - Thermal throttling
- and many more.

2.7.6 M-Box Events Ordered By Code

Table 2-84 summarizes the directly-measured M-Box events.

Table 2-84. Performance Monitor Events for M-Box Events

| Symbol Name | CNT_CTLx[13:9] | SubCtl Dep | Max Inc/ Cyc | Description |
|--|-------------------------------------|-------------|--------------|---------------------------------------|
| Events selected by inc_sel | | | | |
| DSP_FILL | 0x00 | DSP | 1 | Dispatch Queue Events |
| CYCLES_MFULL | 0x01 | | 1 | M-Box Full Cycles |
| RETRY_MFULL | 0x02 | | 1 | Retry MFull |
| RETRY_STARVE | 0x03 | | 1 | Retry Starve |
| THERM_TRP_UP | 0x04 | THR | 1 | DIMM 'Up' Thermal Trip Points Crossed |
| THERM_TRP_DN | 0x05 | THR | 1 | DIMM 'Dn' Thermal Trip Points Crossed |
| REFRESH | 0x06 | | 1 | Refresh Commands |
| REFRESH_CONFLICT | 0x07 | | 1 | Refresh Conflict |
| ISS_SCHED | 0x08 | | 1 | ISS Scheduling |
| ISS_EV | 0x09 | ISS | 1 | ISS Related Events |
| DRAM_CMD | 0x0a | PLD,ISS | 1 | DRAM Commands |
| PLD_RETRY_EV | 0x0b | PLD | 1 | PLD Related Retry Events |
| MA_PAR_ERR | 0x0c | | 1 | MA Parity Error |
| FVC_EV0 | 0x0d | FVC | 1 | FVC Event 0 |
| FVC_EV1 | 0x0e | FVC | 1 | FVC Event 1 |
| FVC_EV2 | 0x0f | FVC | 1 | FVC Event 2 |
| FVC_EV3 | 0x10 | FVC | 1 | FVC Event 3 |
| PATROL_TXNS | 0x11 | | 1 | Patrol Scrub Transaction |
| TRANS_CMDS | 0x12 | | 1 | Translated Commands |
| PAGE_MISS | 0x13 | | 1 | Page Table Misses |
| PAGE_HITS | 0x14 | | 1 | Page Table Hits |
| PGT_PAGE_EV2 | 0x15 | PGT | 1 | PGT Related Page Table Events |
| PGT_PAGE_EV | 0x16 | PGT | 1 | PGT Related Page Table Events |
| MULTICAS | 0x17 | | 1 | Multi-CAS |
| FVID_RACE | 0x18 | | 1 | FVID Race Detected |
| TT_CMD_CONFLICT | 0x19 | | 1 | Thermal Throttling Command Conflicts |
| BBOX_CMDS_ALL | 0x1a | | 1 | All B-Box Commands |
| CYCLES | 0x1b | | 1 | M-Box Cycles |
| SCHED_INFLIGHT_CMDS | 0x1c | | 1 | All B-Box Commands |
| INFLIGHT_CMDS | 0x1d | | 1 | In-flight Commands |
| Symbol Name | CNT_CTLx [21:19] && CNT_CTLx[7] = 1 | Max Inc/Cyc | Max Inc/ Cyc | Description |
| Events selected by set_flag_sel | | | | |
| CYCLES_DSP_FILL | 0x0 | DSP | 1 | Time in DSP_FILL State |

Table 2-84. Performance Monitor Events for M-Box Events

| | | | | |
|-----------------------|-----|-----|---|---|
| CYCLES_SCHED_MODE | 0x1 | ISS | 1 | Time in SCHED_MODE State |
| CYCLES_RETRYQ_MFULL | 0x3 | | 1 | Time RetryQ MFull |
| CYCLES_RETRYQ_STARVED | 0x4 | | 1 | Time RetryQ Starved |
| CYCLES_PGT_STATE | 0x5 | PGT | 1 | Time in Page Table State |
| BCMD_SCHEDQ_OCCUPANCY | 0x6 | | 1 | B-Box Command Scheduler Queue Occupancy |

2.7.7 M-Box Performance Monitor Event List

This section enumerates Intel® Xeon® Processor E7 Family uncore performance monitoring events for the M-Box.

BBOX_CMDS_ALL

- **Title:** All B-Box Commands
- **Category:** M-Box Commands Received
- **Event Code:** 0x1a, **Max. Inc/Cyc:** 1,
- **Definition:** Number of new commands sent from the B-Box and received by the M-Box.

BCMD_SCHEDQ_OCCUPANCY

- **Title:** B-Box Command Scheduler Queue Occupancy
- **Category:** Cycle Events
- **Event Code:** [21:19]0x06 && [7]0x1, **Max. Inc/Cyc:** 1,
- **Definition:** Counts the queue occupancy of the B-Box Command scheduler per FVID. The FVID (Fill Victim Index) for the command to be monitored must be programmed in MSR_PMU_MAP.fvid.

Table 2-85. Unit Masks for BCMD_SCHEDQ_OCCUPANCY

| Extension | MAP[9:6] | Description |
|-----------|----------|---|
| READS | 0x0 | Reads commands to M box from B box |
| WRITES | 0x1 | Write commands from B box to M box |
| MERGE | 0x2 | Merge commands from B box to M box |
| V2F | 0x3 | Victim buffer to Fill buffer transfer (V2F) command from the B-Box to the M-Box |
| V2V | 0x4 | Victim buffer to Victim buffer transfer (V2V) command from the B-Box to the M-Box |
| F2V | 0x5 | Fill buffer to Victim buffer transfer (F2V) command from the B-Box to the M-Box |
| F2B | 0x6 | Fill buffer read to B-Box (F2B) from M-Box |
| SPRWR | 0x7 | spare write commands from the B-Box to the M-Box |
| ALL | 0x8 | All B-Box Commands |

CYCLES

- **Title:** M-Box Cycles
- **Category:** Cycle Events
- **Event Code:** 0x1b, **Max. Inc/Cyc:** 1,
- **Definition:** Count M-box cycles

CYCLES_DSP_FILL

- **Title:** Time in DSP_FILL State
- **Category:** Cycle Events
- **Event Code:** [21:19]0x00 && [7]0x1, **Max. Inc/Cyc:** 1,
- **Definition:** Counts cycles spent in state specified in M_CSR_PMU_DSP register.

Table 2-86. Unit Masks for CYCLES_DSP_FILL

| Extension | DSP[10:7] | Description |
|-----------|-----------|--------------------------------------|
| WRQ_EMPTY | 0x8 | Cycles dispatch write queue is empty |
| RDQ_EMPTY | 0x4 | Cycles dispatch read queue is empty |
| WRQ_FULL | 0x2 | Cycles dispatch write queue is full |
| RDQ_FULL | 0x1 | Cycles dispatch read queue is full |

CYCLES_MFULL

- **Title:** M-Box Full Cycles
- **Category:** Cycle Events
- **Event Code:** 0x01, **Max. Inc/Cyc:** 1,
- **Definition:** Number of cycles spent in the "mfull" state. Also known as the "badly starved" state.

CYCLES_PGT_STATE

- **Title:** Time in Page Table State
- **Category:** Cycle Events
- **Event Code:** [21:19]0x05 && [7]0x1, **Max. Inc/Cyc:** 1,
- **Definition:** Counts cycles page table stays in state as specified in PMU_PGT.openscls_time.

Table 2-87. Unit Masks for CYCLES_PGT_STATE

| Extension | PGT[6] | Description |
|-----------|--------|------------------|
| CLOSED | 0x0 | Closed Page Mode |
| OPEN | 0x1 | Open Page Mode |

CYCLES_RETRYQ_STARVED

- **Title:** Time RetryQ Starved
- **Category:** Cycle Events
- **Event Code:** [21:19]0x4 && [7]0x1, **Max. Inc/Cyc:** 1,
- **Definition:** Counts cycles RetryQ spends in the "starved" state.

CYCLES_RETRYQ_MFULL

- **Title:** Time RetryQ MFull
- **Category:** Cycle Events
- **Event Code:** [21:19]0x3 && [7]0x1, **Max. Inc/Cyc:** 1,
- **Definition:** Counts cycles RetryQ spends in the "mfull" state. Also known as the "badly starved" state.

CYCLES_SCHED_MODE

- **Title:** Time in SCHED_MODE State
- **Category:** Cycle Events
- **Event Code:** [21:19]0x01 && [7]0x1, **Max. Inc/Cyc:** 1,
- **Definition:** Counts cycles spent in scheduling mode specified in M_CSR_PMU_ISS.sched_mode register.

Table 2-88. Unit Masks for CYCLES_SCHED_MODE

| Extension | ISS[6:4] | Description |
|-----------|----------|---|
| TRADEOFF | 0x0 | Trade-off between read latency/write bandwidth. |
| RDPRIO | 0x1 | Minimize read latency. Reads take priority. |
| WRPRIO | 0x2 | Minimize write latency. Writes take priority. |
| ADAPTIVE | 0x3 | Adaptive. |

DRAM_CMD

- **Title:** DRAM Commands
- **Category:** DRAM Commands
- **Event Code:** 0x0a, **Max. Inc/Cyc:** 1,
- **Definition:** Count PLD Related DRAM Events
- **NOTE:** In order to measure a non-filtered version of the subevents, it is necessary to make sure the PLD Dep bits [13,7,0] are also set to 0

| Extension | PLD Dep Bits | ISS Dep Bits | Description |
|-------------------|------------------------|--------------|--|
| ALL | [0]0x0 | | Advance counter when a DRAM command is detected. |
| ILLEGAL | [12:8]0x0 | | Count Invalid DRAM commands. |
| PREALL | [12:8]0x1 | | Count Preall DRAM commands. |
| PREALL.TRDOFF | [12:8]0x1 && [0]0x1 | [9:7]0x0 | Count Preall (no auto-precharge, open page mode) DRAM commands during 'static trade off' scheduling mode. |
| PREALL.RDPRIO | [12:8]0x1 && [0]0x1 | [9:7]0x1 | Count Preall (no auto-precharge, open page mode) DRAM commands during 'static read priority' scheduling mode. |
| PREALL.WRPRIO | [12:8]0x1 && [0]0x1 | [9:7]0x2 | Count Preall (no auto-precharge, open page mode) DRAM commands during 'static write priority' scheduling mode. |
| PREALL.ADAPTIVE | [12:8]0x1 && [0]0x1 | [9:7]0x2 | Count Preall (no auto-precharge, open page mode) DRAM commands during 'adaptive' scheduling mode. |
| RAS | [12:8]0x2 | | Count RAS DRAM commands. |
| RAS.TRDOFF | [12:8]0x2 && [0]0x1 | [9:7]0x0 | Count RAS (no auto-precharge, open page mode) DRAM commands during 'static trade off' scheduling mode. |
| RAS.RDPRIO | [12:8]0x2 && [0]0x1 | [9:7]0x1 | Count RAS (no auto-precharge, open page mode) DRAM commands during 'static read priority' scheduling mode. |
| RAS.WRPRIO | [12:8]0x2 && [0]0x1 | [9:7]0x2 | Count RAS (no auto-precharge, open page mode) DRAM commands during 'static write priority' scheduling mode. |
| RAS.ADAPTIVE | [12:8]0x2 && [0]0x1 | [9:7]0x2 | Count RAS (no auto-precharge, open page mode) DRAM commands during 'adaptive' scheduling mode. |
| CAS_RD_OPN | [12:8]0x3 | | Count CAS Read (no auto-precharge, open page mode) DRAM commands. |
| CAS_WR_OPN | [12:8]0x4 | | Count CAS Write (no auto-precharge, open page mode) DRAM commands. |
| CAS_WR_OPN.TRDOFF | [12:8]0x4 && [0]0x1 | [9:7]0x0 | Count CAS Write (no auto-precharge, open page mode) DRAM commands during 'static trade off' scheduling mode. |
| CAS_WR_OPN.RDPRIO | [12:8]0x4 && [0]0x1 | [9:7]0x1 | Count CAS Write (no auto-precharge, open page mode) DRAM commands during 'static read priority' scheduling mode. |

| Extension | PLD Dep Bits | ISS Dep Bits | Description |
|---------------------|------------------------|--------------|---|
| CAS_WR_OPN.WRPRI0 | [12:8]0x4 && [0]0x1 | [9:7]0x2 | Count CAS Write (no auto-precharge, open page mode) DRAM commands during 'static write priority' scheduling mode. |
| CAS_WR_OPN.ADAPTIVE | [12:8]0x4 && [0]0x1 | [9:7]0x3 | Count CAS Write (no auto-precharge, open page mode) DRAM commands during 'adaptive' scheduling mode. |
| CAS_RD_CLS | [12:8]0x5 | | Count CAS Read (precharge, closed page mode) DRAM commands. |
| CAS_RD_CLS.TRDOFF | [12:8]0x5 && [0]0x1 | [9:7]0x0 | Count CAS Read (precharge, closed page mode) DRAM commands during 'static trade off' scheduling mode. |
| CAS_RD_CLS.RDPRI0 | [12:8]0x5 && [0]0x1 | [9:7]0x1 | Count CAS Read (precharge, closed page mode) DRAM commands during 'static read priority' scheduling mode. |
| CAS_RD_CLS.WRPRI0 | [12:8]0x5 && [0]0x1 | [9:7]0x2 | Count CAS Read (precharge, closed page mode) DRAM commands during 'static write priority' scheduling mode. |
| CAS_RD_CLS.ADAPTIVE | [12:8]0x5 && [0]0x1 | [9:7]0x3 | Count CAS Read (precharge, closed page mode) DRAM commands during 'adaptive' scheduling mode. |
| CAS_WR_CLS | [12:8]0x6 | | Count CAS Write (precharge, closed page mode) DRAM commands. |
| CAS_WR_CLS.TRDOFF | [12:8]0x6 && [0]0x1 | [9:7]0x0 | Count CAS Write (precharge, closed page mode) DRAM commands during 'static trade off' scheduling mode. |
| CAS_WR_CLS.RDPRI0 | [12:8]0x6 && [0]0x1 | [9:7]0x1 | Count CAS Write (precharge, closed page mode) DRAM commands during 'static read priority' scheduling mode. |
| CAS_WR_CLS.WRPRI0 | [12:8]0x6 && [0]0x1 | [9:7]0x2 | Count CAS Write (precharge, closed page mode) DRAM commands during 'static write priority' scheduling mode. |
| CAS_WR_CLS.ADAPTIVE | [12:8]0x6 && [0]0x1 | [9:7]0x3 | Count CAS Write (precharge, closed page mode) DRAM commands during 'adaptive' scheduling mode. |
| MRS | [12:8]0x7 | | Count Mode register set DRAM commands |
| RFR | [12:8]0x9 | | Count Refresh DRAM commands. |
| ENSR | [12:8]0xA | | Count Enter Self-Refresh DRAM commands. |
| EXSR | [12:8]0xB | | Count Exit Self-Refresh DRAM commands. |
| NOP | [12:8]0xC | | Count NOP DRAM commands. |
| TRKL | [12:8]0x10 | | Count Write Trickle DRAM commands. |
| PRE | [12:8]0x11 | | Count PRE DRAM commands. |
| SYNC | [12:8]0x12 | | Count SYNC DRAM commands. |
| CKE_HI | [12:8]0x14 | | Count CKE High DRAM commands. |
| CKE_LO | [12:8]0x15 | | Count CKE Low DRAM commands. |
| SOFT_RST | [12:8]0x17 | | Count Soft Reset DRAM commands. |
| WR_CFG | [12:8]0x1C | | Count Write Configuration Register DRAM commands. |
| RD_CFG | [12:8]0x1D | | Count Read Configuration Register DRAM commands. |
| ZQCAL | [12:8]0x1E | | Count ZQ Calibration DRAM commands. |
| ALL.TRDOFF | [0]0x0 | [9:7]0x0 | Count all DRAM commands during "static trade off" scheduling mode. |
| ALL.RDPRI0 | [0]0x0 | [9:7]0x1 | Count all DRAM commands during "static read priority" scheduling mode. |

| Extension | PLD Dep Bits | ISS Dep Bits | Description |
|------------|--------------|--------------|---|
| ALL.WRPRIO | [0]0x0 | [9:7]0x2 | Count all DRAM commands during "static write priority" scheduling mode. |
| ALL.ADAPT | [0]0x0 | [9:7]0x3 | Count all DRAM commands during "adaptive" scheduling mode. |

DSP_FILL

- **Title:** Dispatch Queue Events
- **Category:** DSP Events
- **Event Code:** 0x00, **Max. Inc/Cyc:** 1,
- **Definition:** Measure a dispatch queue event.

| Extension | DSP[10:7] | Description |
|-----------|-----------|--------------------------------------|
| RDQ_FULL | 0x1 | Cycles dispatch read queue is full |
| WRQ_FULL | 0x2 | Cycles dispatch write queue is full |
| RDQ_EMPTY | 0x4 | Cycles dispatch read queue is empty |
| WRQ_EMPTY | 0x8 | Cycles dispatch write queue is empty |

DRAM_MISC

- **Title:** Miscellaneous DRAM Events
- **Category:** Retry Events
- **Event Code:** 0x0b, **Max. Inc/Cyc:** 1,
- **Definition:** Miscellaneous DRAM Command Events

| Extension | PLD[7] | Description |
|--------------|--------------------------|---|
| RETRIES_ALL | 0x0 | Advance counter when a retry is detected. |
| RETRIES_FVID | 0x1 | Advance counter when a retry to a certain FVID is detected. NOTE: The FVID is programmed into PLD[6:1] |
| Extension | PLD[17] | Description |
| VALID | 0x1 | Count number of Valid commands |
| Extension | PLD[18] | Description |
| NON_NOP_TRKL | 0x1 | Count number of NOP TRKL commands |
| Extension | PLD Dep Bits | Description |
| ILLEGAL | [16]0x1 && [23:19]0x0 | Count Invalid DRAM commands scheduled. |
| PREALL | [16]0x1 && [23:19]0x1 | Count Preall DRAM commands scheduled. |
| RAS | [16]0x1 && [23:19]0x2 | Count RAS DRAM commands scheduled. |
| CAS_RD_OPN | [16]0x1 && [23:19]0x3 | Count CAS Read (no auto-precharge, open page mode) DRAM commands scheduled. |
| CAS_WR_OPN | [16]0x1 && [23:19]0x4 | Count CAS Write (no auto-precharge, open page mode) DRAM commands scheduled. |
| CAS_RD_CLS | [16]0x1 && [23:19]0x5 | Count CAS Read (precharge, closed page mode) DRAM commands scheduled. |

| Extension | PLD[7] | Description |
|------------|---------------------------|--|
| CAS_WR_CLS | [16]0x1 && [23:19]0x6 | Count CAS Write (precharge, closed page mode) DRAM commands scheduled. |
| MRS | [16]0x1 && [23:19]0x7 | Count Mode register set DRAM commands scheduled |
| RFR | [16]0x1 && [23:19]0x9 | Count Refresh DRAM commands scheduled. |
| ENSR | [16]0x1 && [23:19]0xA | Count Enter Self-Refresh DRAM commands scheduled. |
| EXSR | [16]0x1 && [23:19]0xB | Count Exit Self-Refresh DRAM commands scheduled. |
| NOP | [16]0x1 && [23:19]0xC | Count NOP DRAM commands scheduled. |
| TRKL | [16]0x1 && [23:19]0x10 | Count Write Trickle DRAM commands scheduled. |
| PRE | [16]0x1 && [23:19]0x11 | Count PRE DRAM commands scheduled. |
| SYNC | [16]0x1 && [23:19]0x12 | Count SYNC DRAM commands scheduled. |
| CKE_HI | [16]0x1 && [23:19]0x14 | Count CKE High DRAM commands scheduled. |
| CKE_LO | [16]0x1 && [23:19]0x15 | Count CKE Low DRAM commands scheduled. |
| SOFT_RST | [16]0x1 && [23:19]0x17 | Count Soft Reset DRAM commands scheduled. |
| WR_CFG | [16]0x1 && [23:19]0x1C | Count Write Configuration Register DRAM commands scheduled. |
| RD_CFG | [16]0x1 && [23:19]0x1D | Count Read Configuration Register DRAM commands scheduled. |
| ZQCAL | [16]0x1 && [23:19]0x1E | Count ZQ Calibration DRAM commands scheduled. |

FRM_TYPE

- **Title:** Frame (Intel SMI) Types
- **Category:** DRAM Commands
- **Event Code:** 0x09, **Max. Inc/Cyc:** 1,
- **Definition:** Count ISS Related Intel SMI Frame Type Events

| Extension | ISS[3:0] Bits | Description |
|---------------|---------------|--|
| FRM_TYPE_3CMD | 0x0 | Counts 3CMD (3-command) Intel SMI frames |
| FRM_TYPE_WDAT | 0x1 | Counts WDAT (Write Data) Intel SMI frames |
| FRM_TYPE_SYNC | 0x2 | Counts SYNC Intel SMI frames |
| FRM_TYPE_CHNL | 0x3 | Counts CHNL (channel) Intel SMI frames |
| --- | 0x7-0x4 | (*illegal selection*) |
| FRM_TYPE_NOP | 0x8 | Counts nop Intel SMI frames |
| --- | 0xb-0x9 | (*illegal selection*) |
| FRM_TYPE_1CMD | 0xc | Counts all 1CMD (1-command) Intel SMI frames |
| --- | 0xf-0xd | (*illegal selection*) |

FVC_EVO

- **Title:** FVC Event 0
- **Category:** FVC Events
- **Event Code:** 0x0d, **Max. Inc/Cyc:** 1,
- **Definition:** Measure an FVC related event.
- **NOTE:** It is possible to program the FVC register such that up to 4 events from the FVC can be independently monitored. However, the `bcmd_match` and `resp_match` subevents depend on the setting of **additional** bits in the FVC register (11:9 and 8:5 respectively). Therefore, only **ONE** `FVC_EVx.bcmd_match` event may be monitored at any given time. The same holds true for `FVC_EVx.resp_match`

Table 2-89. Unit Masks for FVC_EVO

| Extension | FVC [14:12] | FVC [11:9] | FVC [8:6] | Description |
|------------------------------|-------------|------------|-----------|--|
| SMI_CRC_ERR | 0x0 | | | Count link level Intel SMI CRC errors |
| MEM_ECC_ERR | 0x1 | | | Count memory ECC errors (that is not a link-level CRC error) |
| POISON_TXN | 0x2 | | | Count poison (directory of a write to memory was encoded as poisoned) transactions |
| ALERT_FRAMES | 0x3 | | | Counts alert frames |
| FAST_RESET | 0x4 | | | Fast reset request from M-Boxes |
| BBOX_CMDS.READS ^a | 0x5 | | 0x0 | Reads commands to M box from B box (e.g. reads from memory) |
| BBOX_CMDS.WRITEs | 0x5 | | 0x1 | Write commands from B box to M box (e.g. writes to memory) |
| BBOX_RSP.ACK | 0x6 | 0x0 | | Counts positive acknowledgements. No error was detected. |
| BBOX_RSP.RETRY | 0x6 | 0x1 | | Count Retry Responses. Possibly a correctable error. Retries are generated until it is decided that the error was either correctable or uncorrectable. |
| BBOX_RSP.COR | 0x6 | 0x2 | | Counts corrected (for example, after error trials or just by a retry) |
| BBOX_RSP.UNCOR | 0x6 | 0x3 | | Count Uncorrectable Responses. |
| BBOX_RSP.SPEC_ACK | 0x6 | 0x4 | | Speculative positive acknowledgement for optimized read flow. No error was detected for the transaction. |
| BBOX_RSP.SPR_ACK | 0x6 | 0x5 | | Count positive acknowledgements for command to misbehaving DIMM during sparing. No error was detected for the transaction. |
| --- | 0x6 | 0x6 | | (*nothing will be counted*) |
| BBOX_RSP.SPR_UNCOR | 0x6 | 0x7 | | Counts Uncorrectable responses to B-Box as a result of commands issued to misbehaving DIMM during sparing |
| SMI_NB_TRIG | 0x7 | | | Select Intel SMI Northbound debug event bits from Intel SMI status frames as returned from the Intel (WSM-EX) Scalable Memory Buffers. Used for Debug purposes |

- a. This event includes reads incurred by the patrol scrubber which periodically (at an interval set by the BIOS) reads all lines in memory.

FVC_EV1

- **Title:** FVC Event 1
- **Category:** FVC Events
- **Event Code:** 0x0e, **Max. Inc/Cyc:** 1,
- **Definition:** Measure an FVC related event.
- **NOTE:** It is possible to program the FVC register such that up to 4 events from the FVC can be independently monitored. However, the `bcmd_match` and `resp_match` subevents depend on the setting of **additional** bits in the FVC register (11:9 and 8:5 respectively). Therefore, only **ONE** `FVC_EVx.bcmd_match` event may be monitored at any given time. The same holds true for `FVC_EVx.resp_match`

Table 2-90. Unit Masks for FVC_EV1

| Extension | FVC [17:15] | FVC [11:9] | FVC [8:6] | Description |
|------------------------------|-------------|------------|-----------|--|
| SMI_CRC_ERR | 0x0 | | | Count link level Intel SMI CRC errors |
| MEM_ECC_ERR | 0x1 | | | Count memory ECC errors (that is not a link-level CRC error) |
| POISON_TXN | 0x2 | | | Count poison (directory of a write to memory was encoded as poisoned) transactions |
| ALERT_FRAMES | 0x3 | | | Counts alert frames |
| FAST_RESET | 0x4 | | | Fast reset request from M-Boxes |
| BBOX_CMDS.READS ^a | 0x5 | | 0x0 | Reads commands to M box from B box (e.g. reads from memory) |
| BBOX_CMDS.WRITEs | 0x5 | | 0x1 | Write commands from B box to M box (e.g. writes from memory) |
| BBOX_RSP.ACK | 0x6 | 0x0 | | Counts positive acknowledgements. No error was detected. |
| BBOX_RSP.RETRY | 0x6 | 0x1 | | Count Retry Responses. Possibly a correctable error. Retries are generated until it is decided that the error was either correctable or uncorrectable. |
| BBOX_RSP.COR | 0x6 | 0x2 | | Counts corrected (for example, after error trials or just by a retry) |
| BBOX_RSP.UNCOR | 0x6 | 0x3 | | Count Uncorrectable Responses. |
| BBOX_RSP.SPEC_ACK | 0x6 | 0x4 | | Speculative positive acknowledgement for optimized read flow. No error was detected for the transaction. |
| BBOX_RSP.SPR_ACK | 0x6 | 0x5 | | Count positive acknowledgements for command to misbehaving DIMM during sparing. No error was detected for the transaction. |
| --- | 0x6 | 0x6 | | (*nothing will be counted*) |
| BBOX_RSP.SPR_UNCOR | 0x6 | 0x7 | | Counts Uncorrectable responses to B-Box as a result of commands issued to misbehaving DIMM during sparing |
| SMI_NB_TRIG | 0x7 | | | Select Intel SMI Northbound debug event bits from Intel SMI status frames as returned from the Intel (WSM-EX) Scalable Memory Buffers. Used for Debug purposes |

- a. This event includes reads incurred by the patrol scrubber which periodically (at an interval set by the BIOS) reads all lines in memory.

FVC_EV2

- **Title:** FVC Event 2
- **Category:** FVC Events
- **Event Code:** 0x0f, **Max. Inc/Cyc:** 1,
- **Definition:** Measure an FVC related event.
- **NOTE:** It is possible to program the FVC register such that up to 4 events from the FVC can be independently monitored. However, the `bcmd_match` and `resp_match` subevents depend on the setting of **additional** bits in the FVC register (11:9 and 8:5 respectively). Therefore, only **ONE** `FVC_EVx.bcmd_match` event may be monitored at any given time. The same holds true for `FVC_EVx.resp_match`

Table 2-91. Unit Masks for FVC_EV2

| Extension | FVC [20:18] | FVC [11:9] | FVC [8:6] | Description |
|------------------------------|-------------|------------|-----------|--|
| SMI_CRC_ERR | 0x0 | | | Count link level Intel SMI CRC errors |
| MEM_ECC_ERR | 0x1 | | | Count memory ECC errors (that is not a link-level CRC error) |
| POISON_TXN | 0x2 | | | Count poison (directory of a write to memory was encoded as poisoned) transactions |
| ALERT_FRAMES | 0x3 | | | Counts alert frames |
| FAST_RESET | 0x4 | | | Fast reset request from M-Boxes |
| BBOX_CMDS.READS ^a | 0x5 | | 0x0 | Reads commands to M box from B box (e.g. reads from memory) |
| BBOX_CMDS.WRITEs | 0x5 | | 0x1 | Write commands from B box to M box (e.g. writes from memory) |
| BBOX_RSP.ACK | 0x6 | 0x0 | | Counts positive acknowledgements. No error was detected. |
| BBOX_RSP.RETRY | 0x6 | 0x1 | | Count Retry Responses. Possibly a correctable error. Retries are generated until it is decided that the error was either correctable or uncorrectable. |
| BBOX_RSP.COR | 0x6 | 0x2 | | Counts corrected (for example, after error trials or just by a retry) |
| BBOX_RSP.UNCOR | 0x6 | 0x3 | | Count Uncorrectable Responses. |
| BBOX_RSP.SPEC_ACK | 0x6 | 0x4 | | Speculative positive acknowledgement for optimized read flow. No error was detected for the transaction. |
| BBOX_RSP.SPR_ACK | 0x6 | 0x5 | | Count positive acknowledgements for command to misbehaving DIMM during sparing. No error was detected for the transaction. |
| --- | 0x6 | 0x6 | | (*nothing will be counted*) |
| BBOX_RSP.SPR_UNCOR | 0x6 | 0x7 | | Counts Uncorrectable responses to B-Box as a result of commands issued to misbehaving DIMM during sparing |
| SMI_NB_TRIG | 0x7 | | | Select Intel SMI Northbound debug event bits from Intel SMI status frames as returned from the Intel (WSM-EX) Scalable Memory Buffers. Used for Debug purposes |

- a. This event includes reads incurred by the patrol scrubber which periodically (at an interval set by the BIOS) reads all lines in memory.

FVC_EV3

- **Title:** FVC Event 3
- **Category:** FVC Events
- **Event Code:** 0x10, **Max. Inc/Cyc:** 1,
- **Definition:** Measure an FVC related event.
- **NOTE:** It is possible to program the FVC register such that up to 4 events from the FVC can be independently monitored. However, the `bcmd_match` and `resp_match` subevents depend on the setting of **additional** bits in the FVC register (11:9 and 8:5 respectively). Therefore, only **ONE** `FVC_EVx.bcmd_match` event may be monitored at any given time. The same holds true for `FVC_EVx.resp_match`

Table 2-92. Unit Masks for FVC_EV3

| Extension | FVC [23:21] | FVC [11:9] | FVC [8:6] | Description |
|------------------------------|-------------|------------|-----------|--|
| SMI_CRC_ERR | 0x0 | | | Count link level Intel SMI CRC errors |
| MEM_ECC_ERR | 0x1 | | | Count memory ECC errors (that is not a link-level CRC error) |
| POISON_TXN | 0x2 | | | Count poison (directory of a write to memory was encoded as poisoned) transactions |
| ALERT_FRAMES | 0x3 | | | Counts alert frames |
| FAST_RESET | 0x4 | | | Fast reset request from M-Boxes |
| BBOX_CMDS.READS ^a | 0x5 | | 0x0 | Reads commands to M box from B box (e.g. reads from memory) |
| BBOX_CMDS.WRITEs | 0x5 | | 0x1 | Write commands from B box to M box (e.g. writes from memory) |
| BBOX_RSP.ACK | 0x6 | 0x0 | | Counts positive acknowledgements. No error was detected. |
| BBOX_RSP.RETRY | 0x6 | 0x1 | | Count Retry Responses. Possibly a correctable error. Retries are generated until it is decided that the error was either correctable or uncorrectable. |
| BBOX_RSP.COR | 0x6 | 0x2 | | Counts corrected (for example, after error trials or just by a retry) |
| BBOX_RSP.UNCOR | 0x6 | 0x3 | | Count Uncorrectable Responses. |
| BBOX_RSP.SPEC_ACK | 0x6 | 0x4 | | Speculative positive acknowledgement for optimized read flow. No error was detected for the transaction. |
| BBOX_RSP.SPR_ACK | 0x6 | 0x5 | | Count positive acknowledgements for command to misbehaving DIMM during sparing. No error was detected for the transaction. |
| --- | 0x6 | 0x6 | | (*nothing will be counted*) |
| BBOX_RSP.SPR_UNCOR | 0x6 | 0x7 | | Counts Uncorrectable responses to B-Box as a result of commands issued to misbehaving DIMM during sparing |
| SMI_NB_TRIG | 0x7 | | | Select Intel SMI Northbound debug event bits from Intel SMI status frames as returned from the Intel (WSM-EX) Scalable Memory Buffers. Used for Debug purposes |

- a. This event includes reads incurred by the patrol scrubber which periodically (at an interval set by the BIOS) reads all lines in memory.

FVID_RACE

- **Title:** FVID Race Detected
- **Category:** Misc
- **Event Code:** 0x18, **Max. Inc/Cyc:** 1,
- **Definition:** Number of FVID (Fill Victim Index) races detected.
- **NOTE:** This is a race condition where an IMT entry is recycled prematurely. It should not be observable in hardware.

INFLIGHT_CMDS

- **Title:** In-flight Commands
- **Category:** M-Box Commands Received
- **Event Code:** 0x1d, **Max. Inc/Cyc:** 1,
- **Definition:** Number of new memory controller (read and write) commands accepted

ISS_SCHED

- **Title:** ISS Scheduling
- **Category:** M-Box Commands Received
- **Event Code:** 0x08, **Max. Inc/Cyc:** 1,
- **Definition:** ISS Scheduling

| Extension | ISS Bits[10] | Description |
|------------|--------------|---|
| CHANGES | 0x0 | Number of ISS scheduling mode transitions detected. |
| FRAME_BEAT | 0x1 | Number of Frame Beats |

MA_PAR_ERR

- **Title:** MA Parity Error
- **Category:** Misc
- **Event Code:** 0x0c, **Max. Inc/Cyc:** 1,
- **Definition:** Number of MA even parity errors detected.

MULTICAS

- **Title:** Multi-CAS
- **Category:** Misc
- **Event Code:** 0x17, **Max. Inc/Cyc:** 1,
- **Definition:** Number of Multi-CAS patrol transactions detected. This is an indication of multiple CAS commands to the same open page, and should correlate to page hit rate.

PAGE_HIT

- **Title:** Page Table Hit
- **Category:** Page Table Related
- **Event Code:** 0x14, **Max. Inc/Cyc:** 1,
- **Definition:** Number of page hits detected. This is a command that requires a only a CAS to complete.

PAGE_MISS

- **Title:** Page Table Misses
- **Category:** Page Table Related
- **Event Code:** 0x13, **Max. Inc/Cyc:** 1,
- **Definition:** Number of page misses detected. This is a command that requires a PRE-RAS-CAS to complete.

PATROL_TXNS

- **Title:** Patrol Scrub Transaction
- **Category:** Misc
- **Event Code:** 0x11, **Max. Inc/Cyc:** 1,
- **Definition:** Number of patrol scrub transactions detected.

PGT_PAGE_EV

- **Title:** PGT Related Page Table Events
- **Category:** Page Table Related
- **Event Code:** 0x16, **Max. Inc/Cyc:** 1,
- **Definition:** Counts PGT Related Page Table Events.

| Extension | PGT Bits[0] | Description |
|-----------|-------------|--|
| OPN2CLS | 0x0 | Advance counter when an open-to-closed page mode transition is detected. |
| CLS2OPN | 0x1 | Advance counter when an closed-to-open page mode transition is detected. |

PGT_PAGE_EV2

- **Title:** PGT Related Page Table Events
- **Category:** Page Table Related
- **Event Code:** 0x15, **Max. Inc/Cyc:** 1,
- **Definition:** Counts PGT Related Page Table Events.

| Extension | PGT Bits[11] | Description |
|------------|--------------|--|
| AUTO_CLS | 0x0 | Number of page auto close commands. |
| PAGE_EMPTY | 0x1 | Number of empty page hits. Number of transactions accessing a bank with no open page. The page was previously closed either because it has never been opened, was closed via a CASpre, closed explicitly by the idle page closing mechanism, or closed by a PREALL in order to do a refresh. This is command that requires a RAS-CAS to complete. |

REFRESH

- **Title:** Refresh Commands
- **Category:** DRAM Commands
- **Event Code:** 0x06, **Max. Inc/Cyc:** 1,
- **Definition:** Advance counter when a refresh command is detected.

REFRESH_CONFLICT

- **Title:** Refresh Conflict
- **Category:** DRAM Commands
- **Event Code:** 0x07, **Max. Inc/Cyc:** 1,
- **Definition:** Number of refresh conflicts detected. A refresh conflict is a conflict between a read/write transaction and a refresh command to the same rank.

RETRY_MFULL

- **Title:** Retry MFull
- **Category:** Retry Events
- **Event Code:** 0x02, **Max. Inc/Cyc:** 1,
- **Definition:** Number of retries detected while in the "mfull" state. Also known as the "badly starved" state.

RETRY_STARVE

- **Title:** Retry Starve
- **Category:** Retry Events
- **Event Code:** 0x03, **Max. Inc/Cyc:** 1,
- **Definition:** Number of retries detected while in the "starved" state.

SCHED_INFLIGHT_CMDS

- **Title:** Scheduler In-flight Commands
- **Category:** M-Box Commands Received
- **Event Code:** 0x1c, **Max. Inc/Cyc:** 1,
- **Definition:** Number of new scheduler commands that were accepted.

THERM_TRP_DN

- **Title:** DIMM `Dn` Thermal Trip Points Crossed
- **Category:** Thermal Throttle
- **Event Code:** 0x05, **Max. Inc/Cyc:** 1,
- **Definition:** Counts when a specified thermal trip point is crossed in the "down" direction.

| Extension | THR Bits [10:9],[3] | Description |
|---------------------|---------------------|--|
| ALL.GT_MID_RISE | 0x3,0x1 | Advance the counter when the above mid temp thermal trip point (rising) is crossed in the "down" direction for any DIMM |
| ALL.GT_MID_FALL | 0x2,0x1 | Advance the counter when the above mid temp thermal trip point (falling) is crossed in the "down" direction for any DIMM |
| ALL.GT_LO | 0x1,0x1 | Advance the counter when the above low temp, but below mid temp thermal trip point is crossed in the "down" direction for any DIMM. |
| ALL.LT_LO | 0x0,0x1 | Advance the counter when the below low temp thermal trip point is crossed in the "down" direction for any DIMM |
| DIMM{n}.GT_MID_RISE | 0x3,0x0 | Advance the counter when the above mid temp thermal trip point (rising) is crossed in the "down" direction for DIMM #. NOTE: THR Bits [6:4] must be programmed with the DIMM # |
| DIMM{n}.GT_MID_FALL | 0x2,0x0 | Advance the counter when the above mid temp thermal trip point (falling) is crossed in the "down" direction for DIMM #. NOTE: THR Bits [6:4] must be programmed with the DIMM # |
| DIMM{n}.GT_LO | 0x1,0x0 | Advance the counter when the above low temp, but below mid temp thermal trip point is crossed in the "down" direction for DIMM #. NOTE: THR Bits [6:4] must be programmed with the DIMM # |
| DIMM{n}.LT_LO | 0x0,0x0 | Advance the counter when the below low temp, but below mid temp thermal trip point is crossed in the "down" direction for DIMM #. NOTE: THR Bits [6:4] must be programmed with the DIMM # |

THERM_TRP_UP

- **Title:** DIMM 'Up' Thermal Trip Points Crossed
- **Category:** Thermal Throttle
- **Event Code:** 0x04, **Max. Inc/Cyc:** 1,
- **Definition:** Counts when a specified thermal trip point is crossed in the "up" direction.

| Extension | THR Bits [8:7],[3] | Description |
|---------------------|--------------------|--|
| ALL.GT_MID_RISE | 0x3,0x1 | Advance the counter when the above mid temp thermal trip point (rising) is crossed in the "up" direction for any DIMM |
| ALL.GT_MID_FALL | 0x2,0x1 | Advance the counter when the above mid temp thermal trip point (falling) is crossed in the "up" direction for any DIMM |
| ALL.GT_LO | 0x1,0x1 | Advance the counter when the above low temp, but below mid temp thermal trip point is crossed in the "up" direction for any DIMM. |
| ALL.LT_LO | 0x0,0x1 | Advance the counter when the below low temp thermal trip point is crossed in the "up" direction for any DIMM |
| DIMM{n}.GT_MID_RISE | 0x3,0x0 | Advance the counter when the above mid temp thermal trip point (rising) is crossed in the "up" direction for DIMM #. NOTE: THR Bits [6:4] must be programmed with the DIMM # |
| DIMM{n}.GT_MID_FALL | 0x2,0x0 | Advance the counter when the above mid temp thermal trip point (falling) is crossed in the "up" direction for DIMM #. NOTE: THR Bits [6:4] must be programmed with the DIMM # |
| DIMM{n}.GT_LO | 0x1,0x0 | Advance the counter when the above low temp, but below mid temp thermal trip point is crossed in the "up" direction for DIMM #. NOTE: THR Bits [6:4] must be programmed with the DIMM # |
| DIMM{n}.LT_LO | 0x0,0x0 | Advance the counter when the below low temp, but below mid temp thermal trip point is crossed in the "up" direction for DIMM #. NOTE: THR Bits [6:4] must be programmed with the DIMM # |

TRANS_CMDS

- **Title:** Translated Commands
- **Category:** Dispatch Queue
- **Event Code:** 0x12, **Max. Inc/Cyc:** 1,
- **Definition:** Counts read/write commands entered into the Dispatch Queue.

TT_CMD_CONFLICT

- **Title:** Thermal Throttling Command Conflicts
- **Category:** Thermal Throttle
- **Event Code:** 0x19, **Max. Inc/Cyc:** 1,
- **Definition:** Count command conflicts due to thermal throttling.

2.8 W-Box Performance Monitoring

2.8.1 Overview of the W-Box

The W-Box is the primary Power Controller for the Intel® Xeon® Processor E7 Family.

2.8.2 W-Box Performance Monitoring Overview

The W-Box supports event monitoring through four 48-bit wide counters (W_MSR_PERF_CNT{3:0}). Each of these four counters can be programmed to count any W-Box event. The W-Box counters will increment by a maximum of 1 per cycle.

The W-Box also provides a 48-bit wide fixed counter that increments at the uncore clock frequency.

For information on how to setup a monitoring session, refer to [Section 2.1, "Global Performance Monitoring Control Counter Overflow"](#).

- Umask enables core to scope events for each core (one-hot encoding) (Bit 0 in umask for Core 0, Bit 1 in umask for core 1, etc.).

2.8.2.1 W-Box PMU - Overflow, Freeze and Unfreeze

If an overflow is detected from a W-Box performance counter, the overflow bit is set at the box level (W_MSR_PMON_GLOBAL_STATUS.ov/ov_fixed), and forwarded to the U-Box where it is also captured (U_MSR_PMON_GLOBAL_STATUS.ov_w).

HW can be also configured (by setting the corresponding *.pmi_en* to 1) to send a PMI to the U-Box when an overflow is detected. The U-Box may be configured to freeze all uncore counting and/or send a PMI to selected cores when it receives this signal.

Once a freeze has occurred, in order to see a new freeze, the overflow field responsible for the freeze, must be cleared by setting the corresponding bit in W_MSR_PMON_GLOBAL_OVF_CTL.clr_ov/clr_ov_fixed. Assuming all the counters have been locally enabled (.en bit in data registers meant to monitor events) and the overflow bit(s) has been cleared, the W-Box is prepared for a new sample interval. Once the global controls have been re-enabled ([Section 2.1.5, "Enabling a New Sample Interval from Frozen Counters"](#)), counting will resume.

2.8.3 W-BOX Performance Monitors

Table 2-93. W-Box Performance Monitoring MSRs

| MSR Name | Access | MSR Address | Size (bits) | Description |
|--------------------------------|--------|-------------|-------------|------------------------------------|
| W_MSR_PMON_FIXED_CTL_CTL | RW_RW | 0x395 | 64 | W-Box PMON Fixed Counter Control |
| W_MSR_PMON_FIXED_COUNTER | RW_RW | 0x394 | 64 | W-Box PMON Fixed Counter |
| | | | | |
| W_MSR_PMON_COUNTER_3 | RW_RW | 0xC97 | 64 | W-Box PMON Counter 3 |
| W_MSR_PMON_CTRL_3 | RW_RW | 0xC96 | 64 | W-Box PMON Control 3 |
| W_MSR_PMON_COUNTER_2 | RW_RW | 0xC95 | 64 | W-Box PMON Counter 2 |
| W_MSR_PMON_CTRL_2 | RW_RW | 0xC94 | 64 | W-Box PMON Control 2 |
| W_MSR_PMON_COUNTER_1 | RW_RW | 0xC93 | 64 | W-Box PMON Counter 1 |
| W_MSR_PMON_CTRL_1 | RW_RW | 0xC92 | 64 | W-Box PMON Control 1 |
| W_MSR_PMON_COUNTER_0 | RW_RW | 0xC91 | 64 | W-Box PMON Counter 0 |
| W_MSR_PMON_CTRL_0 | RW_RW | 0xC90 | 64 | W-Box PMON Control 0 |
| | | | | |
| W_MSR_PMON_GLOBAL_OVERFLOW_CTL | RW_RW | 0xC82 | 32 | W-Box PMON Global Overflow Control |
| W_MSR_PMON_GLOBAL_STATUS | RW_RW | 0xC81 | 32 | W-Box PMON Global Overflow Status |
| W_MSR_PMON_GLOBAL_CTL | RW_RW | 0xC80 | 32 | W-Box PMON Global Control |

2.8.3.1 W-Box Box Level PMON state

The following registers represent the state governing all box-level PMUs in the W-Box.

The `_GLOBAL_CTL` register contains the bits used to enable monitoring. It is necessary to set the `.ctr_en/fixed_en` bit to 1 before the corresponding data register can collect events.

If an overflow is detected from one of the W-Box PMON registers, the corresponding bit in the `_GLOBAL_STATUS.ov/ov_fixed` field will be set. To reset the overflow bits set in the `_GLOBAL_STATUS.ov/ov_fixed` field, a user must set the corresponding bits in the `_GLOBAL_OVF_CTL.clr_ov/clr_ov_fixed` field before beginning a new sample interval.

Table 2-94. W_MSR_PMON_GLOBAL_CTL Register Fields

| Field | Bits | HW Reset Value | Description |
|----------|------|----------------|--|
| fixed_en | 31 | 0 | Enable the fixed counter |
| ig | 30:4 | 0 | Read zero; writes ignored. |
| ctr_en | 3:0 | 0 | Must be set to enable each WBOX counter (bit 0 to enable ctr0, etc) NOTE: U-Box enable and per counter enable must also be set to fully enable the counter. |

Table 2-95. W_MSR_PMON_GLOBAL_STATUS Register Fields

| Field | Bits | HW Reset Value | Description |
|----------|------|----------------|--|
| ov_fixed | 31 | 0 | If an overflow is detected from the WBOX PMON fixed counter, this bit will be set. |
| ig | 30:4 | 0 | Read zero; writes ignored. |
| ov | 3:0 | 0 | If an overflow is detected from the corresponding WBOX PMON register, it's overflow bit will be set. |

Table 2-96. W_MSR_PMON_GLOBAL_OVF_CTRL Register Fields

| Field | Bits | HW Reset Value | Description |
|--------------|------|----------------|---|
| clr_ov_fixed | 31 | 0 | Writing '1' to bit in field causes corresponding bit in 'Overflow PerfMon Counter' field in W_MSR_PMON_GLOBAL_STATUS register to be cleared to 0. |
| ig | 30:4 | 0 | Read zero; writes ignored. |
| clr_ov | 5:0 | 0 | Writing '1' to bit in field causes corresponding bit in 'Overflow PerfMon Counter' field in W_MSR_PMON_GLOBAL_STATUS register to be cleared to 0. |

2.8.3.2 W-Box PMON state - Counter/Control Pairs

The following table defines the layout of the W-Box performance monitor control registers. The main task of these configuration registers is to select the event to be monitored by their respective data counter. Setting the `.ev_sel` and `.umask` fields performs the event selection. The `.en` bit must be set to 1 to enable counting.

Additional control bits include:

- `.pmi_enable` governs what to do if an overflow is detected.
- `.threshold` - since C-Box counters can increment by a value greater than 1, a threshold can be applied. If the `.threshold` is set to a non-zero value, that value is compared against the incoming count for that event in each cycle. If the incoming count is \geq the threshold value, then the event count captured in the data register will be incremented by 1. (Not present in fixed counter)
- `.invert` - Changes the `.threshold` test condition to '<' (Not present in fixed counter)
- `.edge_detect` - Rather than accumulating the raw count each cycle (for events that can increment by 1 per cycle), the register can capture transitions from no event to an event incoming. (Not present in fixed counter)

Table 2-97. W_MSR_PMON_EVT_SEL_{3-0} Register – Field Definitions

| Field | Bits | HW Reset Value | Description |
|-------------|-------|----------------|---|
| ig | 63 | 0 | Read zero; writes ignored. |
| rsv | 62:61 | 0 | Reserved; Must write to 0 else behavior is undefined. |
| ig | 60:51 | 0 | Read zero; writes ignored. |
| rsv | 50 | 0 | Reserved; Must write to 0 else behavior is undefined. |
| ig | 49:32 | 0 | Read zero; writes ignored. |
| thresh | 31:24 | 0 | Threshold used for counter comparison. |
| invert | 23 | 0 | Invert threshold comparison. When '0', the comparison will be thresh >= event. When '1', the comparison will be threshold < event. |
| en | 22 | 0 | Counter enable |
| ig | 21 | 0 | Read zero; writes ignored. |
| pmi_en | 20 | 0 | PMI Enable. If bit is set, when corresponding counter overflows, a PMI exception is sent to the U-Box. |
| ig | 19 | 0 | Read zero; writes ignored. |
| edge_detect | 18 | 0 | Edge Detect. When bit is set, 0->1 transition of a one bit event input will cause counter to increment. When bit is 0, counter will increment for however long event is asserted. |
| ig | 17:16 | 0 | Read zero; writes ignored. |
| umask | 15:8 | 0 | In W-Box, this field is used to enable core scope events per core. Bit 0 masks Core 0, bit 1 masks Core 1, etc. |
| ev_sel | 7:0 | 0 | Event Select |

Table 2-98. W_MSR_PMON_FIXED_CTR_CTL Register – Field Definitions

| Field | Bits | HW Reset Value | Description |
|--------|------|----------------|--|
| ig | 63:3 | 0 | Read zero; writes ignored. |
| rsv | 2 | 0 | Reserved; Must write to 0 else behavior is undefined. |
| pmi_en | 1 | 0 | PMI Enable. If bit is set, when corresponding counter overflows, a PMI exception is sent to the U-Box. |
| en | 0 | 0 | Counter enable |

The W-Box performance monitor data registers are 48b wide. A counter overflow occurs when a carry out bit from bit 47 is detected. Software can force all uncore counting to freeze after N events by preloading a monitor with a count value of $(2^{48} - 1) - N$ and setting the control register to send a PMI to the U-Box. Upon receipt of the PMI, the U-Box will disable counting ([Section 2.1.1, "Freezing on Counter Overflow"](#)). During the interval of time between overflow and global disable, the counter value will wrap and continue to collect events.

In this way, software can capture the precise number of events that occurred between the time uncore counting was enabled and when it was disabled (or 'frozen') with minimal skew.

If accessible, software can continuously read the data registers without disabling event collection.

Table 2-99. W_MSR_PMON_CTR_{3-0} Register – Field Definitions

| Field | Bits | HW Reset Value | Description |
|-------------|------|----------------|----------------------------------|
| event_count | 47:0 | 0 | 48-bit performance event counter |

Table 2-100. W_MSR_PMON_FIXED_CTR Register – Field Definitions

| Field | Bits | HW Reset Value | Description |
|-------------|------|----------------|----------------------------------|
| event_count | 47:0 | 0 | 48-bit performance event counter |

Note: Due to an errata found within the Intel® Xeon® Processor E7 Family, SW must consider two special cases:

- If SW reads a counter whose value ends in 0x000000 or 0x000001, SW should subtract 0x1000000 to get the correct value.
- SW should not set up a sample interval whose value ends in either 0xfffffe or 0xfffff.

2.8.4 W-BOX Performance Monitoring Events

2.8.4.1 An Overview:

The W-Box's primary offering to understanding the impact of the uncore on performance is the fixed counter. This counter, which increments at the frequency of the uncore clock, can be used to add a time element to numerous events across the uncore.

Beyond that, the W-Box provides a smattering of events that indicating when, and under what circumstances, the W-Box throttled the chip due to power constraints.

2.8.5 W-Box Events Ordered By Code

Table 2-101 summarizes the directly-measured W-Box events.

Table 2-101. Performance Monitor Events for W-Box Events

| Symbol Name | Event Code | Max Inc/ Cyc | Description |
|---------------------|------------|--------------|-------------------------------------|
| C_THROTTLE_TMP | 0x00 | 1 | Core Throttled due to Temp |
| C_CO_THROTTLE_DIE | 0x01 | 1 | Core Throttled in CO |
| PROCHOT | 0x02 | 1 | Prochot |
| CO_THROTTLE_PROCHOT | 0x03 | 1 | Core Throttled in CO due to FORCEPR |
| C_CYCLES_TURBO | 0x04 | 1 | Core in CO at Turbo |
| TM1_ON | 0x07 | 1 | TM1 Throttling On |
| RATIO_CHANGE_ABORT | 0x08 | 1 | Ratio Change Abort |

2.8.6 W-Box Performance Monitor Event List

This section enumerates Intel Xeon Processor (WSM-EX) Series uncore performance monitoring events for the W-Box.

C_CYCLES_TURBO

- **Title:** Core in C0 at Turbo
- **Category:** W-Box Events
- **Event Code:** 0x04, **Max. Inc/Cyc:** 1,
- **Definition:** Selected core is in C0 and operating at a 'Turbo' operating point.

C_C0_THROTTLE_DIE

- **Title:** Core Throttled in C0
- **Category:** W-Box Events
- **Event Code:** 0x01, **Max. Inc/Cyc:** 1,
- **Definition:** Selected core is in C0 and is throttling.

C_C0_THROTTLE_PROCHOT

- **Title:** Core Throttled in C0 due to FORCEPR
- **Category:** W-Box Events
- **Event Code:** 0x03, **Max. Inc/Cyc:** 1,
- **Definition:** Selected core is in C0 and is throttling due to FORCEPR assertion.

C_THROTTLE_TMP

- **Title:** Core Throttled due to Temp
- **Category:** W-Box Events
- **Event Code:** 0x00, **Max. Inc/Cyc:** 1,
- **Definition:** Temperature of the selected core is at or above the throttle temperature.

PROCHOT

- **Title:** Prochot
- **Category:** W-Box Events
- **Event Code:** 0x02, **Max. Inc/Cyc:** 1,
- **Definition:** Package is asserting the PROCHOT output.

RATIO_CHANGE_ABORT

- **Title:** Ratio Change Abort
- **Category:** W-Box Events
- **Event Code:** 0x08, **Max. Inc/Cyc:** 1,
- **Definition:** Selected core aborted a ratio change request.

TM1_ON

- **Title:** TM1 Throttling On
- **Category:** W-Box Events
- **Event Code:** 0x07, **Max. Inc/Cyc:** 1,
- **Definition:** Selected core is in C0 and TM1 throttling is occurring.

2.9 Packet Matching Reference

In several boxes (S, R and B), the performance monitoring infrastructure allows a user to filter packet traffic according to certain fields. A couple common fields, the Message Class/Opcode fields, have been summarized in the following tables.

Table 2-102. Intel® QuickPath Interconnect Packet Message Classes

| Code | Name | Definition |
|-------|------|-----------------------|
| b0000 | HOM0 | Home - Requests |
| b0001 | HOM1 | Home - Responses |
| b0010 | NDR | Non-Data Responses |
| b0011 | SNP | Snoops |
| b0100 | NCS | Non-Coherent Standard |
| --- | | |
| b1100 | NCB | Non-Coherent Bypass |
| --- | | |
| b1110 | DRS | Data Response |

Table 2-103. Opcode Match by Message Class

| Opc | HOMO | HOM1 | SNP | DRS |
|------------|-----------------|-------------|--------------|---------------------------|
| 0000 | RdCur | Rspl | SnpCur | DataC_(FEIMS) |
| 0001 | RdCode | RspS | SnpCode | DataC_(FEIMS)_FrcAckCnflt |
| 0010 | RdData | --- | SnpData | DataC_(FEIMS)_Cmp |
| 0011 | NonSnpRd | --- | --- | DataNc |
| 0100 | RdInvOwn | RspCnflt | SnpInvOwn | WblData |
| 0101 | InvXtol | --- | SnpInvXtol | WbSData |
| 0110 | EvctCln | RspCnfltOwn | --- | WbEData |
| 0111 | NonSnpWr | --- | --- | NonSnpWrData |
| 1000 | InvltoE | RspFwd | SnpInvltoE | WblDataPtl |
| 1001 | AckCnfltWbl | RspFwdl | --- | --- |
| 1010 | --- | RspFwdS | --- | WbEDataPtl |
| 1011 | --- | RspFwdlWb | --- | NonSnpWrdataPtl |
| 1100 | WbMtol | RspFwdSWb | --- | --- |
| 1101 | WbMtoE | RsplWb | --- | --- |
| 1110 | WbMtoS | RspSWb | --- | --- |
| 1111 | AckCnflt | --- | PrefetchHint | --- |
| Opc | NDR | NCB | NCS | |
| 0000 | Gnt_Cmp | NcWr | NcRd | |
| 0001 | Gnt_FrcAckCnflt | WcWr | IntAck | |
| 0010 | --- | --- | --- | |
| 0011 | --- | --- | FERR | |
| 0100 | CmpD | --- | NcRdPtl | |
| 0101 | AbortTO | --- | NcCfgRd | |
| 0110 | --- | --- | --- | |
| 0111 | --- | --- | NcIORd | |
| 1000 | Cmp | NcMsgB | --- | |
| 1001 | FrcAckCnflt | IntLogical | NcCfgWr | |
| 1010 | Cmp_FwdCode | IntPhysical | --- | |
| 1011 | Cmp_FwdInvOwn | IntPrioUpd | NcIOWr | |
| 1100 | Cmp_FwdInvltoE | NcWrPtl | NcMsgS | |
| 1101 | --- | WcWrPtl | NcP2PS | |
| 1110 | --- | NcP2PB | --- | |
| 1111 | --- | DebugData | --- | |

Table 2-104. Opcodes (Alphabetical Listing)

| Name | Opc | MC | Gen By? | Desc |
|---------------------------|------|------|---------|--|
| AbortTO | 0101 | NDR | | Abort Time-out Response |
| AckCnflt | 1111 | HOMO | | Acknowledge receipt of Data_* and Cmp/FrcAckCnflt, signal a possible conflict scenario. |
| AckCnfltWbl | 1001 | HOMO | | In addition to signaling AckCnflt, the caching agent has also written the dirty cache line data plus any partial write data back to memory in a WBiData[PtI] message and transitioned the cache line state to I. |
| Cmp | 1000 | NDR | Uo | All snoop responses gathered, no conflicts |
| CmpD | 0100 | NDR | Uo | Completion with Data |
| Cmp_FwdCode | 1010 | NDR | | Complete request, forward the line in F (or S) state to the requestor specified, invalidate local copy or leave it in S state. |
| Cmp_FwdInvtoE | 1100 | NDR | | Complete request, invalidate local copy |
| Cmp_FwdInvOwn | 1011 | NDR | | Complete request, forward the line in E or M state to the requestor specified, invalidate local copy |
| DataC_(FEIMS) | 0000 | DRS | | Data Response in (FEIMS) state NOTE: Set RDS field to specify which state is to be measured. - Intel® Xeon® Processor E7 Family supports getting data in E, F, I or M state |
| DataC_(FEIMS)_Cmp | 0010 | DRS | | Data Response in (FEIMS) state, Complete NOTE: Set RDS field to specify which state is to be measured. - Intel® Xeon® Processor E7 Family supports getting data in E, F or I state |
| DataC_(FEIMS)_FrcAckCnflt | 0001 | DRS | | Data Response in (FEIMS) state, Force Acknowledge NOTE: Set RDS field to specify which state is to be measured. - Intel® Xeon® Processor E7 Family supports getting data in E, F or I state |
| DataNc | 0011 | DRS | Uo | Non-Coherent Data |
| DebugData | 1111 | NCB | | Debug Data |
| EvctCln | 0110 | HOMO | | Clean cache line eviction notification to home agent. |
| FERR | 0011 | NCS | | Legacy floating point error indication from CPU to legacy bridge |
| FrcAckCnflt | 1001 | NDR | | All snoop responses gathered, force an AckCnflt |
| Gnt_Cmp | 0000 | NDR | | Signal completion and Grant E state ownership without data to an InvtoE or 'null data' to an InvXtol |
| Gnt_FrcAckCnflt | 0001 | NDR | | Signal FrcAckCnflt and Grant E state ownership without data to an InvtoE or 'null data' to an InvXtol |
| IntAck | 0001 | NCS | | Interrupt acknowledge to legacy 8259 interrupt controller |
| IntLogical | 1001 | NCB | Ui, Uo | Logical mode interrupt to processor |
| IntPhysical | 1010 | NCB | Ui, Uo | Physical mode interrupt to processor |
| IntPrioUpd | 1011 | NCB | Ui, Uo | Interrupt priority update message to source interrupt agents. |
| InvtoE | 1000 | HOMO | | Invalidate to E state requests exclusive ownership of a cache line without data. |

| Name | Opc | MC | Gen By? | Desc |
|-----------------|------|------|---------|--|
| InvXtol | 0101 | HOMO | | Flush a cache line from all caches (that is, downgrade all clean copies to I and cause any dirty copy to be written back to memory). |
| NcCfgRd | 0101 | NCS | Ui | Configuration read from configuration space |
| NcCfgWr | 1001 | NCS | Ui | Configuration write to configuration space |
| NcIORd | 0111 | NCS | | Read from legacy I/O space |
| NcIOWr | 1011 | NCS | | Write to legacy I/O space |
| NcMsgB | 1000 | NCB | Ui, Uo | Non-coherent Message (non-coherent bypass channel) |
| NcMsgS | 1100 | NCS | Ui, Uo | Non-coherent Message (Non-coherent standard channel) |
| NcP2PB | 1110 | NCB | | Peer-to-peer transaction between I/O entities (non-coherent bypass channel) |
| NcP2PS | 1101 | NCS | | Peer-to-peer transaction between I/O entities. (Non-coherent standard channel) |
| NcRd | 0000 | NCS | Ui | Read from non-coherent memory mapped I/O space |
| NcRdPtl | 0100 | NCS | Ui | Partial read from non-coherent memory mapped I/O space |
| NcWr | 0000 | NCB | | Write to non-coherent memory mapped I/O space |
| NcWrPtl | 1100 | NCB | Ui | Partial write to non-coherent memory mapped I/O space |
| NonSnpRd | 0011 | HOMO | | Non-Snoop (uncached) read |
| NonSnpWr | 0111 | HOMO | | Non-Snoop (uncached) write |
| NonSnpWrData | 0111 | DRS | | Non cache coherent write data |
| NonSnpWrDataPtl | 1011 | DRS | | Partial (byte-masked) non cache coherent write data |
| PrefetchHint | 1111 | SNP | | Snoop Prefetch Hint |
| RdCode | 0001 | HOMO | | Read cache line in F (or S, if the F state not supported) |
| RdCur | 0000 | HOMO | | Request a cache line in I. Typically issued by I/O proxy entities, RdCur is used to obtain a coherent snapshot of an uncached cache line. |
| RdData | 0010 | HOMO | | Read cache line in either E or F (or S, if F state not supported). The choice between F (or S) and E is determined by whether or not per caching agent has cache line in S state. |
| RdInvOwn | 0100 | HOMO | | Read Invalidate Own requests a cache line in M or E state. M or E is determined by whether requester is forwarded an M copy by a peer caching agent or sent an E copy by home agent. |
| RspCnflt | 0100 | HOM1 | | Peer is left with line in I or S state, and the peer has a conflicting outstanding request. |
| RspCnfltOwn | 0110 | HOM1 | | Peer has a buried M copy for this line with an outstanding conflicting request. |
| RspFwd | 1000 | HOM1 | | Peer has sent data to requestor with no change in cache state |
| RspFwdI | 1001 | HOM1 | | Peer has sent data to requestor and is left with line in I state |
| RspFwdIWb | 1011 | HOM1 | | Peer has sent data to requestor and a WbIData to the home, and is left with line in I state |
| RspFwdS | 1010 | HOM1 | | Peer has sent data to requestor and is left with line in S state |
| RspFwdSWb | 1100 | HOM1 | | Peer has sent data to requestor and a WbSData to the home, and is left with line in S state |

| Name | Opc | MC | Gen By? | Desc |
|--------------|------|------|---------|---|
| Rspl | 0000 | HOM1 | | Peer left with line in I-state |
| RsplWb | 1101 | HOM1 | | Peer has evicted the data with an in-flight WblData[Ptl] message to the home and has not sent any message to the requestor. |
| RspS | 0001 | HOM1 | | Peer left with line in S-state |
| RspSwb | 1110 | HOM1 | | Peer has sent a WbSData message to the home, has not sent any message to the requestor and is left with line in S-state |
| SnpcCode | 0001 | SNP | | Snoop Code (get data in F or S state) - Intel® Xeon® Processor E7 Family supports getting data in F state |
| Snpcur | 0000 | SNP | | Snoop to get data in I state |
| SnpcData | 0010 | SNP | | Snoop Data (get data in E, F or S state) - Intel® Xeon® Processor E7 Family supports getting data in E or F state |
| SnpcInvltoE | 1000 | SNP | | Snoop Invalidate to E state. To invalidate peer caching agent, flushing any M state data to home |
| SnpcInvlOwn | 0100 | SNP | | Snoop Invalidate Own (get data in E or M state) - Intel® Xeon® Processor E7 Family supports getting data in E state |
| SnpcInvlXtol | 0101 | SNP | | Snoop Invalidate Writeback M to I state. To invalidate peer caching agent, flushing any M state data to home. |
| WbEData | 0110 | DRS | | Writeback data, downgrade to E state |
| WbEDataPtl | 1010 | DRS | | Partial (byte-masked) writeback data, downgrade to E state |
| WblData | 0100 | DRS | | Writeback data, downgrade to I state |
| WblDataPtl | 1000 | DRS | | Partial (byte-masked) writeback data, downgrade to I state |
| WbMtoI | 1100 | HOMO | | Write a cache line in M state back to memory and transition its state to I. |
| WbMtoE | 1101 | HOMO | | Write a cache line in M state back to memory and transition its state to E. |
| WbMtoS | 1110 | HOMO | | Write a cache line in M state back to memory and transition its state to S. |
| WbSData | 0101 | DRS | | Writeback data, downgrade to S state |
| WcWr | 0001 | NCB | | Write combinable write to non-coherent memory mapped I/O space |
| WcWrPtl | 1101 | NCB | | Partial write combinable write to non-coherent memory mapped I/O space |