5th Gen Intel® Xeon® Scalable Processor XCC (Codename Emerald Rapids) Uncore Performance

Monitoring Guide

August 2024

Revision 001

Reference Number: 817509

Contents

Figures

Tables

Revision History

Revision Number	Description	Date
001	• Initial Release.	August 2024

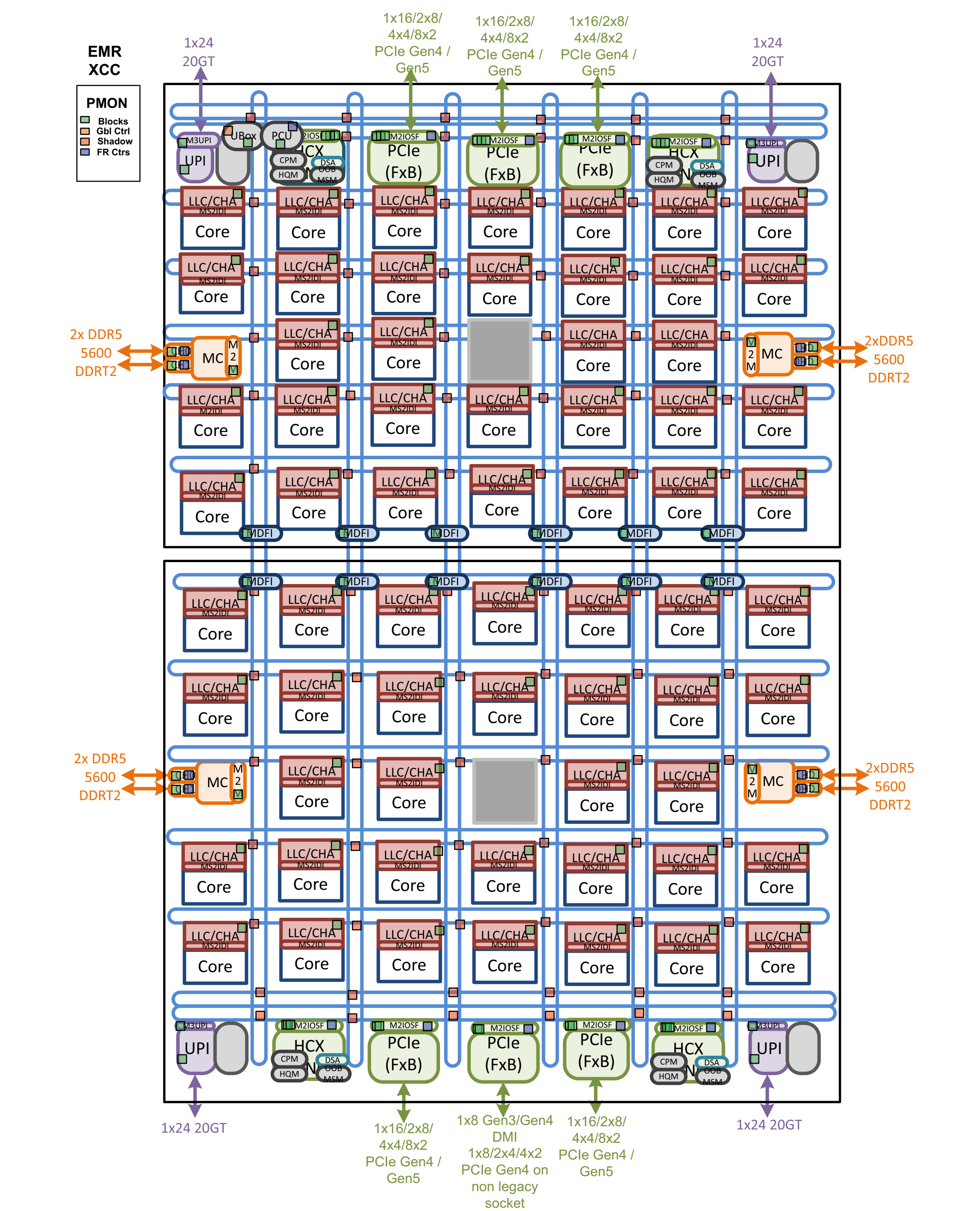
1 Introduction

1.1 Introduction

'Uncore' roughly equates to logic outside the CPU cores but residing on the same die. Traffic (for example, data reads) generated by threads executing on CPU cores or IO devices may be operated on by logic in the uncore. Logic responsible for managing coherency, managing access to the DIMMs, managing power distribution and sleep states, and so forth.

The uncore sub-system of the next generation Intel® Xeon® Server Processor is shown in Figure 1-1. The uncore sub-system consists of a variety of components, many assigned to the aforementioned responsibilities, ranging from the CHA cache/home agent to the Power Controller Unit (PCU) and Integrated Memory Controller (IMC), to name a few. Most of these components provide similar performance monitoring capabilities.

Figure 1-1. 5th Gen Intel® Xeon® Scalable Processor XCC Block Diagram



Before going in to the details of 5th Gen Intel® Xeon® Scalable Processor's uncore PMON, the following sections will provide:

- A general overview of Uncore PMON operation and the state provided SW to manage its operation.
- Functionality common to individual units with the common logic to support the functionality.
- A summary of 5th Gen Intel® Xeon® Scalable Processor uncore performance monitoring capabilities.
- Addressing all 5th Gen Intel® Xeon® Scalable Processor uncore performance monitoring state.
- Introduction to new discovery mechanism.
- Some guidance to SW including how to manage a monitoring session, find the base address to the page of Discovery and find the base addresses for PMON registers addressed in PCICFG or MMIO space.

1.2 Section References

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

- Section 2.1, "Mesh Performance Monitoring".
- Section 2.2, "Caching/Home Agent (CHA) Performance Monitoring".
- Section 2.3, "Memory Controller (iMC) Performance Monitoring".
- Section 2.4, "IIO Performance Monitoring".
- Section 2.5, "IIO Ring Port (IRP) Performance Monitoring".
- Section 2.6, "Intel® UPI Link Layer Performance Monitoring".
- Section 2.7, "M2M Performance Monitoring".
- Section 2.8, "M2PCIe* Performance Monitoring".
- Section 2.9, "M3UPI Performance Monitoring".
- Section 2.10, "Power Control (PCU) Performance Monitoring".
- Section 2.11, "MDF Performance Monitoring".
- Section 2.12, "Compute Express Link* Performance Monitoring".

1.3 Uncore PMON Overview

1.3.1 A Simple Hierarchy

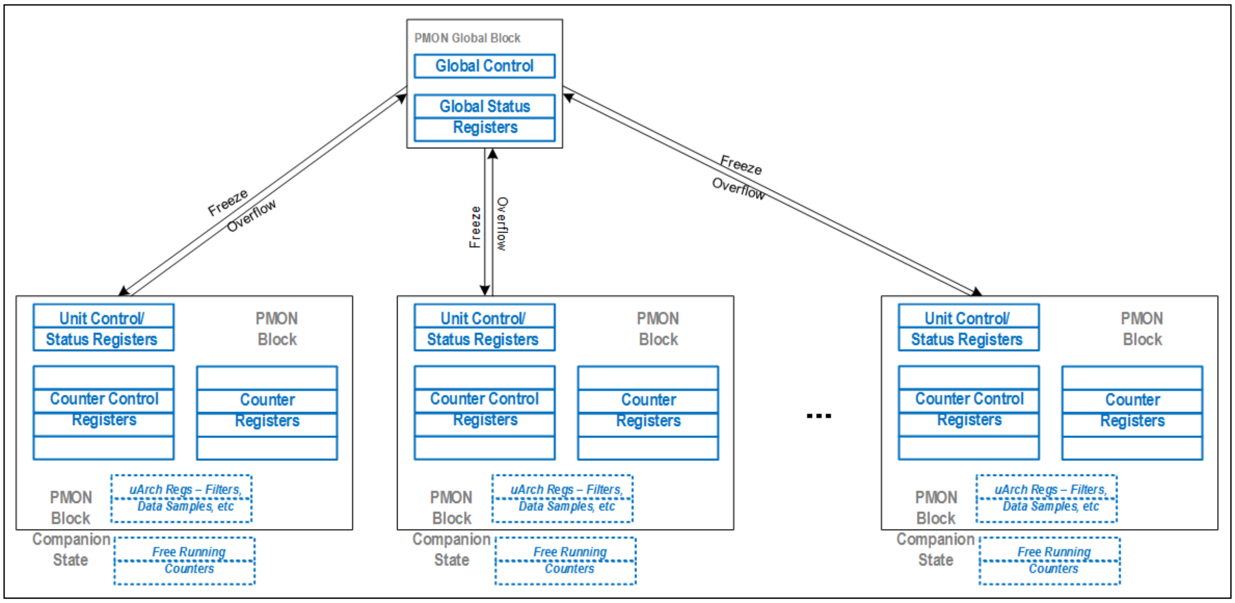
Uncore performance monitoring is managed through a very simple hierarchy. There are some number of Performance Monitoring (or '**PMON**') units governed by a global control.

Each PMON block contains a set of counters with paired control registers. Each unit provides a set of events for the SW to select from. The SW can ask the HW to collect an event by specifying what to count in a counter's control register. The SW can then periodically read the collected value from the paired counter.

Some units offer an expanded event set that require additional counter control bits. (for example, CHA, IIO, and Intel® Ultra Path Interconnect (Intel® UPI)).

Some units offer the ability to further refine, or 'filter', the monitored events through additional counter control registers.

Figure 1-2. Uncore PMON Components and Hierarchy



Note: Uncore performance monitors represent a per-socket resource not meant to be affected by context switches and thread migration performed by the OS. It is recommended that the monitoring software agent establish a fixed affinity binding to prevent event count cross-talk across uncore PMON collected from different sockets.

To manage the large number of counter registers distributed across so many units and collect event data efficiently, each block has a modest amount of control/status governed by a similar global control/status.

The SW can directly synchronize actions across counters (for example, to start/stop/ reset counting) within each PMON block or across all PMON blocks through this control state.

The SW can indirectly synchronize actions across counters (for example, stop counting) in all the PMON blocks by telling the HW what to do when a counter overflows. The SW can set a counter to overflow, after a set number of events have been captured, by pre-seeding the counter. For each counter, the SW can then choose whether to notify the global PMON control that a counter has overflowed.

Upon receipt of an overflow, the global control will assert the global freeze signal. Once the global freeze has been detected, each box will disable (or 'freeze') all of its counters. In the process of generating a global freeze, SW can configure the global control to send a PMI signal to the core executing the monitoring software.

The following sections will detail the basic control state provided to SW to control performance monitoring in the uncore.

1.3.2 Global PMON State

1.3.2.1 Global PMON Global Control/Status Registers

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

U_MSR_PMON_GLOBAL_CTL contains *bits that can stop (.frz_all) all the uncore* counters.

Figure 1-3. PMON Global Control Register for 5th Gen Intel® Xeon® Scalable Processor

Table 1-1. U_MSR_PMON_GLOBAL_CTL Register – Field Definitions

Field	Bits	Attr	HW Reset Val	Description
rsv	60:1	RV	0	Reserved
frz_all	0	WO	0	Freeze all uncore performance monitors.

If an overflow is detected in any of the uncore's PMON registers, it will be summarized in one or more U_MSR_PMON_GLOBAL_STATUS registers. These registers accumulate overflows sent to it from uncore boxes with PMON blocks. To reset these overflow bits, a user must set the corresponding bits in U_MSR_PMON_GLOBAL_STATUS to 1, which will act to clear them.

Figure 1-4. PMON Global Status Register for 5th Gen Intel® Xeon® Scalable Processor

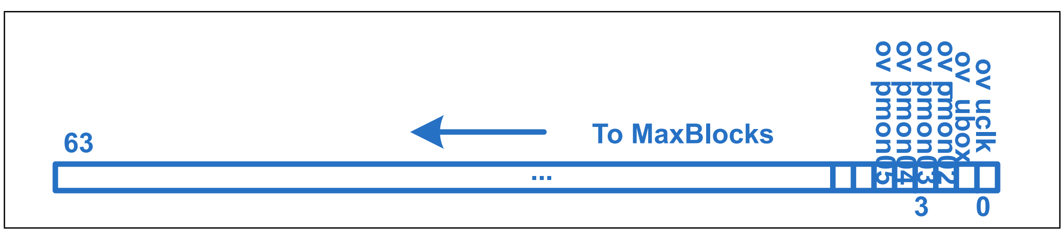


Table 1-2. U_MSR_PMON_GLOBAL_STATUS Register – Field Definitions

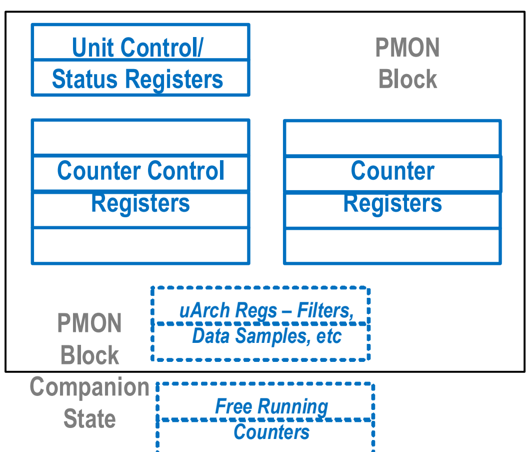
Field	Bits	Attr	HW Reset Val	Description
rsv	63:Max Blocks	RV	0	Reserved
ov_pmonX	MaxBlo cks-1:4	RW1C	0	Overflow detected in PMON register from Block with "Global Status Position" of "MaxBlocks-1" as reported through Global Discovery.
ov_pmonx-1: ov_pmon04	MaxBlo cks-2:4	RW1C	0	Overflow detected in PMON register(s) from Blocks with a Global Status Position between MaxBlocks-1 and 3
ov_pmon03	3	RW1C	0	Overflow detected in PMON register from Block with "Global Status Position" of 3
ov_pmon02	2	RW1C	0	Overflow detected in PMON register from Block with "Global Status Position" of 2

The mapping of Global Status bits in the Global Status register(s) to PMON blocks will be provided through the new PMON discovery mechanism. The status bits correspond to overflow's detected from PMON Block's IDed through discovery. Discovery for each PMON block will report its "Global Status Position" (that is, which bit in the global status register records its overflows).

For instance, the SW may discover a PMON block of Unit Type = CHA, Unit ID 5 has a Global Status Position of 5.

1.4 Unit Level PMON State

Each PMON block in the uncore is composed of the following state:



- A Unit Control register to aid software sample collection.
- Status registers to record when a counter within the Block overflows.
- A set of data registers

- A set of control registers, each paired to a data register, to allow the SW to specify what event should be captured.
- Additional micro-architectural specific state designed to enhance performance monitoring collection within a block. For example, event or traffic filters.
- Some free running counters, although not subject to the PMON hierarchy, may be included in this document with the unit they are associated with.

Every PMON block in the system is governed by a modest amount of unit-level control. Each bit intended to assist the SW in more efficiently managing the PMON state within the block. Reset bits help reduce the time the SW needs to setup a new sample.

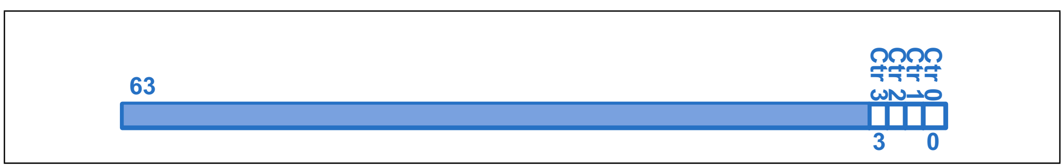
Note: If the PMON registers within the unit are shared among different users, either those users should leave this register untouched or they should agree on the user allowed to affect the unit level control state.

Figure 1-5. PMON Unit Control Register for 5th Gen Intel® Xeon® Scalable Processor - Common to all PMON Blocks

Table 1-3. PMON_UNIT_CTL Register – Field Definitions

Field	Bits	Attr	HW Reset Val	Description
rst_ctrs	9	WO	0	Reset Counters. When set to 1, the Counter Registers will be reset to 0.
rst_ctrl	8	WO	0	Reset Control. When set to 1, the Counter Control Registers will be reset to 0.
rsv	7:1	RV	0	Reserved
frz	0	WO	0	Freeze. If set to 1 the counters in this box will be frozen.

Figure 1-6. PMON Unit Status Register for 5th Gen Intel® Xeon® Scalable Processor - Format Common to all PMON Blocks



If an overflow is detected from one of the unit's PMON registers, the corresponding bit in the *PMON_UNIT_STATUS.ov* field will be set. To reset these overflow bits, a user must write a value of '1' to them (which will clear the bits). There are typically four counters per PMON block. But that number may vary. As of the 5th Gen Intel® Xeon® Scalable Processor, the number of paired counter/counter control registers is reported through the unit discovery associated with each PMON block. The Unit Status register will contain "NumControlRegs" valid bits.

Note: You can also check Table 1-7, "Per-Box Performance Monitoring Capabilities" or the section detailing each unit's functionality for the number counters it supports.

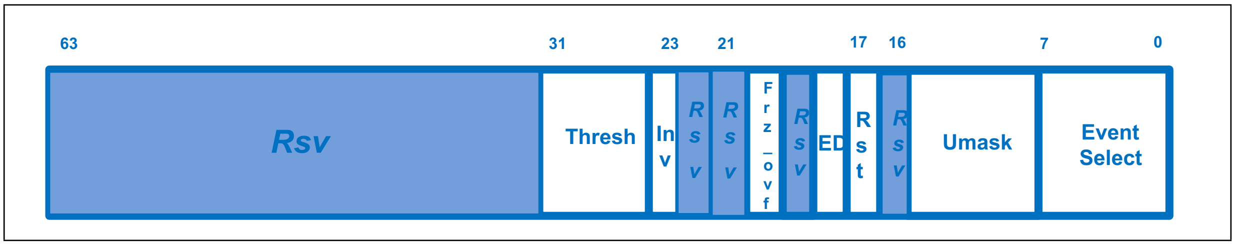
Table 1-4. PMON_UNIT_STATUS Register – Field Definitions

Field	Bits	Attr	HW Reset Val	Description
rsv	31:4	RV	0	Reserved
ov	NumControlRegs-1:0	RW1C	0	If an overflow is detected from the corresponding PMON_CTR register, its overflow bit will be set. **Note:** Write of '1' will clear the bit. Although four is very common, the number of overflow bits can vary by PMON block. The number can be discovered in the NumControlRegs field of the unit's discovery.

1.4.0.1 Unit PMON state - Counter/Control Pairs

The following table defines the layout for the standard performance monitor control registers. Their main task is to select the event to be monitored by their respective data counter (*.ev_sel*, *.umask*). Additional control bits are provided to shape the incoming events (for example, *.invert*, *.edge_det*, *.thresh*) as well as provide additional functionality for monitoring software (*.rst*).

Figure 1-7. PMON Counter Control Register for 5th Gen Intel® Xeon® Scalable Processor - Fields common to all PMON Blocks



Note: Per Unit considerations - refer to each unit's section for more detail on:

- Certain units may make use of additional bits in these counter control registers.
- The width of the Thresh field is dependent on a unit's 'widest' event (that is, the event that can increment the most per cycle, typically measuring per-cycle occupancy of a large queue).
- Several unit counter control registers are still 32b, some 64b. All are addressable as 64b registers.

An overview of the counter control logic is in the next section.

Table 1-5. Baseline *_PMON_CTLx Register – Field Definitions

Field	Bits	Attr	HW Reset Val	Description
rsv	63:32	RV	0	Reserved - Only relevant to unit’s that use 64b control registers
thresh	31:24	RW	0	Threshold is used, along with the invert bit, to compare against the counter’s incoming increment value. That is, the value that will be added to the counter. For events that increment by more than 1 per cycle, if the threshold is set to a value greater than 1, the data register will accumulate instances in which the event increment is ≥ threshold. For example, say you have an event to accumulate the occupancy of a 64-entry queue every cycle. By setting the threshold value to 60, the data register would count the number of cycles the queue’s occupancy was ≥ 60.
invert(inv)	23	RW	0	Invert comparison against Threshold. 0 - Comparison will be ‘is event increment >= threshold?’ 1 - Comparison is inverted - ‘is event increment < threshold?’ For example, for a 64-entry queue, if the SW wanted to know how many cycles the queue had fewer than four entries, the SW should set the threshold to 4 and set the invert bit to 1. **Note:** *.invert* is in series following *.thresh*. Due to this, the *.thresh* field must be set to a non-0 value. For events that increment by no more than 1 per cycle, set *.thresh* to 0x1. Also, if *.edge_det* is set to 1, the counter will increment when a 1 to 0 transition (that is, falling edge) is detected.
rsv	22:21	RV	0	Reserved. SW must write to 0 else behavior is undefined.
frz_ovf	20	RW/V	0	Freeze on overflow When an overflow is detected from this register, a PMON overflow message is sent to the global control. This bit will tell the global control whether it should assert the global freeze for all the counters in the same domain.
rsv	19	RV	0	Reserved.
edge_det(ED)	18	RW	0	When set to 1, rather than measuring the event in each cycle it is active, the corresponding counter will increment when a 0 to 1 transition (that is, rising edge) is detected. When 0, the counter will increment in each cycle that the event is asserted. **Note:** *.edge_det* is in series following *.thresh*. Due to this, the *.thresh* field must be set to a non-0 value. For events that increment by no more than 1 per cycle, set *.thresh* to 0x1.
rst	17	WO	0	When set to 1, the corresponding counter will be cleared to 0.
rsv	16	RV	0	Reserved. SW must write to 0 else behavior is undefined.
umask	15:8	RW	0	Select sub-events to be counted within the selected event.
event select	7:0	RW	0	Select event to be counted.

The default width for performance monitor data registers are 48b wide. A counter overflow occurs when a carry out from bit 47 is detected. Software can force all uncore counting to freeze after N events by pre-loading a monitor with a count value of 2^{48} - N and setting the control register to send an overflow message to the UBox (refer to Section 1.3.2, “Global PMON State”). During the interval of time between overflow and global disable, the counter value will wrap and continue to collect events.

To ensure accuracy, the SW should stop the counter and check the overflow status before reading its value. But, if accessible, the SW can continuously read the data registers without disabling event collection.

Figure 1-8. PMON Counter Register for 5th Gen Intel® Xeon® Scalable Processor - Common to all PMON Blocks

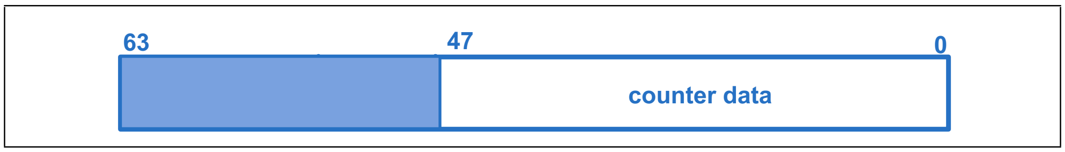


Table 1-6. Baseline *_PMON_CTRx Register – Field Definitions

Field	Bits	Attr	HW Reset Val	Description
rsv	63:48	RV	0	Reserved
event_count	47:0	RW-V	0	48-bit performance event counter

1.4.0.2 Unit PMON Registers - On Overflow and the Consequences (PMI/Freeze)

If an overflow is detected from a unit’s performance counter, the overflow bit is set at the unit level (*_PMON_UNIT_STATUS.ov).

If the counter is enabled to communicate the overflow (*_PMON_CTL.frz_on_en is set to 1), an overflow message is sent to the UBox. When the UBox receives the overflow signal, the *_PMON_GLOBAL_STATUS.ov_x bit is set, a global freeze signal is sent and a PMI can be generated. ‘x’ represents the box generating the overflow (see Table 1-2, “U_MSR_PMON_GLOBAL_STATUS Register – Field Definitions”).

Once a freeze has occurred, in order to see a new freeze, the overflow responsible for the freeze must be cleared by setting the corresponding bit in *_PMON_UNIT_STATUS.ov and U_MSR_PMON_GLOBAL_STATUs.ov_x to 1 (which acts to clear the bits).

Assuming all counters have been locally enabled (the *.en* bit is set to 1 in every control register meant to monitor events) and the overflow bits have been cleared, the unit is prepared for a new sample interval. Once the global controls have been re-enabled (Section 1.10.6, “Enabling a New Sample Interval from Frozen Counters”), counting will resume.

1.5 Uncore PMON - Typical Counter Control Logic

Selecting What To Monitor: The main task of a configuration register is to select the event to be monitored by its respective data counter. Setting the *.ev_sel* and *.umask* fields performs the event selection.

Applying a Threshold to Incoming Events: *.thresh* - since most counters can increment by a value greater than 1, a threshold can be applied to generate an event based on the outcome of the comparison. If *.thresh* 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 ≥ the threshold value, then the event count captured in the data register will be incremented by 1.

Using the threshold field to generate additional events can be particularly useful when applied to a queue occupancy count. For example, if a queue is known to contain eight entries, it may be useful to know how often it contains six or more entries (that is, almost full) or when it contains one or more entries (that is, not empty).

Note: For the 5th Gen Intel® Xeon® Scalable Processor, the *.invert* and *.edge_det* bits follow the threshold comparison in sequence. If a user wishes to apply these bits to events that only increment by 1 per cycle, *thresh* must be set to 0x1.

Inverting the Threshold Comparison: *.invert* - Changes *.thresh* test condition to '<'.

Counting State Transitions Instead of per-Cycle Events: *.edge_det* - 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 (that is, the 'Rising Edge').

1.6 Uncore PMON - Typical Counter Logic

Telling the HW to enable counting on the Control Register: The *event select bit* must be 1 (clock ticks event) or greater to enable counting. Once the Control register has been configured (refer to Section 1.10.4, "Setting up a Monitoring Session" for more information), the paired data register will begin to collect events.

Notification after X events: *.frz_on_ovf* - Instead of manually stopping the counters at intervals (often wall clock time) pre-determined by software, the hardware can be set to notify the monitoring software when a set number of events has occurred. The *Overflow Enable* bit is provided for just that purpose. See Section 1.4.0.2, "Unit PMON Registers - On Overflow and the Consequences (PMI/Freeze)" for more information on how to use this mechanism.

1.7 5th Gen Intel® Xeon® Scalable Processor's Uncore PMON

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

Table 1-7. Per-Box Performance Monitoring Capabilities

Box	# Boxes	# Counters/ Box	Packet Match/ Mask Filters?	Bit Width
CHA	*up to 64*	*4*	Y	48
IIO	*up to 10 for XCC*	*4 (+1) per stack (+4 per port)*	N	48
IRP	*up to 10 for XCC*	*2*	N	48
IMC	*up to 4 (each with up to 2 channels)*	*4*	N	48
Intel® UPI	*up to 4 links*	*4 (per link)*	Y	48
M3UPI	*up to 4 links*	*4 (per link)*	N	48
M2M	*up to 4*	*4*	Y	48
M2PCIe*	*up to 10 for XCC*	*4*	N	48
PCU	*1*	*4 (+2)*	N	48
CXL.CM/DP	*up to 6*	*8/4*	Y	48
MDF	*up to 14*	*4*	N	48

The programming interface of the counter registers and control registers fall into three address spaces:

- CHA, M2PCIe*, IIO, IRP, PCU, and U-Box PMON registers are accessed through x86 RD/WRMSR instructions. See Table 1-9, "Uncore Performance Monitoring Registers (MSR)".
- iMC PMON registers are accessed through MMIO address space. M2M, Intel UPI, and M3UPI PMON registers are accessed through PCI device configuration space. See Table 1-12, "Uncore Performance Monitoring Registers (PCICFG)" for details.

Irrespective of the address-space difference and with only minor exceptions, the bit-granular layout of the control registers to program event code, unit mask, start/stop, and signal filtering via threshold/edge detect are the same.

1.7.1 Querying Number of CHAs

Note: The number of CHAs varies with the number of cores in a system. To determine the number of CHAs, SW should read bits 31:0 in the CAPID6 register located at Device 30, Function 3, Offset 0x9C and CAPID7 register located at Offset 0xA0.

1.7.2 Querying Number of Intel® UPI Links

The number of Intel UPI links varies according to the specific version of the product. To determine the number of Intel UPI links, SW should read bits 23:28 in the CAPID2 register located at Device 30, Function 3, Offset 0x94.

1.8 Addressing Uncore PMON State

Following is a list of registers provided in the 5th Gen Intel® Xeon® Scalable Processor Uncore for Performance Monitoring. The registers are split between MSR space and PCICFG space.

1.8.1 Uncore Performance Monitoring State in MSR Space

First off, the Global Control / Status Registers.

Table 1-8. Global Performance Monitoring Registers (MSR)

MSR Addresses	Description
0x2FF0	Global Control
0x2FF2,0x2FF3	Global Status
0x2FDE	UCLK Fixed Counter Control
0x2FDF	UCLK Fixed Counter

As mentioned previously, PMON blocks in the uncore have some number of paired counter/control (typically four) registers, a unit status and unit control register. Many units may offer extra PMON state such as event filters or fixed counters.

The addresses for all basic PMON state addressed through MSR space are laid out in the next table.

Table 1-9. Uncore Performance Monitoring Registers (MSR) (Sheet 1 of 4)

Unit	Unit Status	Unit Ctrl	Ctr3	Ctr2	Ctr1	Ctr0	Ctrl3	Ctrl2	Ctrl1	Ctrl0	Extra
CHA											Filter0
CHA 0	0x2001	0x2000	0x200B	0x200A	0x2009	0x2008	0x2005	0x2004	0x2003	0x2002	0x200E
CHA 1	0x2011	0x2010	0x201B	0x201A	0x2019	0x2018	0x2015	0x2014	0x2013	0x2012	0x201E
CHA 2	0x2021	0x2020	0x202B	0x202A	0x2029	0x2028	0x2025	0x2024	0x2023	0x2022	0x202E
CHA 3	0x2031	0x2030	0x203B	0x203A	0x2039	0x2038	0x2035	0x2034	0x2033	0x2032	0x203E
CHA 4	0x2041	0x2040	0x204B	0x204A	0x2049	0x2048	0x2045	0x2044	0x2043	0x2042	0x204E
CHA 5	0x2051	0x2050	0x205B	0x205A	0x2059	0x2058	0x2055	0x2054	0x2053	0x2052	0x205E
CHA 6	0x2061	0x2060	0x206B	0x206A	0x2069	0x2068	0x2065	0x2064	0x2063	0x2062	0x206E
CHA 7	0x2071	0x2070	0x207B	0x207A	0x2079	0x2078	0x2075	0x2074	0x2073	0x2072	0x207E
CHA 8	0x2081	0x2080	0x208B	0x208A	0x2089	0x2088	0x2085	0x2084	0x2083	0x2082	0x208E
CHA 9	0x2091	0x2090	0x209B	0x209A	0x2099	0x2098	0x2095	0x2094	0x2093	0x2092	0x209E
CHA 10	0x20A1	0x20A0	0x20AB	0x20AA	0x20A9	0x20A8	0x20A5	0x20A4	0x20A3	0x20A2	0x20AE
CHA 11	0x20B1	0x20B0	0x20BB	0x20BA	0x20B9	0x20B8	0x20B5	0x20B4	0x20B3	0x20B2	0x20BE
CHA 12	0x20C1	0x20C0	0x20CB	0x20CA	0x20C9	0x20C8	0x20C5	0x20C4	0x20C3	0x20C2	0x20CE
CHA 13	0x20D1	0x20D0	0x20DB	0x20DA	0x20D9	0x20D8	0x20D5	0x20D4	0x20D3	0x20D2	0x20DE
CHA 14	0x20E1	0x20E0	0x20EB	0x20EA	0x20E9	0x20E8	0x20E5	0x20E4	0x20E3	0x20E2	0x20EE
CHA 15	0x20F1	0x20F0	0x20FB	0x20FA	0x20F9	0x20F8	0x20F5	0x20F4	0x20F3	0x20F2	0x20FE
CHA 16	0x2101	0x2100	0x210B	0x210A	0x2109	0x2108	0x2105	0x2104	0x2103	0x2102	0x210E

Table 1-9. Uncore Performance Monitoring Registers (MSR) (Sheet 2 of 4)

Unit	Unit Status	Unit Ctrl	Ctr3	Ctr2	Ctr1	Ctr0	Ctrl3	Ctrl2	Ctrl1	Ctrl0	Extra
CHA 17	0x2111	0x2110	0x211B	0x211A	0x2119	0x2118	0x2115	0x2114	0x2113	0x2112	0x211E
CHA 18	0x2121	0x2120	0x212B	0x212A	0x2129	0x2128	0x2125	0x2124	0x2123	0x2122	0x212E
CHA 19	0x2131	0x2130	0x213B	0x213A	0x2139	0x2138	0x2135	0x2134	0x2133	0x2132	0x213E
CHA 20	0x2141	0x2140	0x214B	0x214A	0x2149	0x2148	0x2145	0x2144	0x2143	0x2142	0x214E
CHA 21	0X2151	0X2150	0X215B	0X215A	0X2159	0X2158	0X2155	0X2154	0X2153	0X2152	0X215E
CHA 22	0x2161	0x2160	0x216B	0x216A	0x2169	0x2168	0x2165	0x2164	0x2163	0x2162	0x216E
CHA 23	0x2171	0x2170	0x217B	0x217A	0x2179	0x2178	0x2175	0x2174	0x2173	0x2172	0x217E
CHA 24	0x2181	0x2180	0x218B	0x218A	0x2189	0x2188	0x2185	0x2184	0X2183	0x2182	0x218E
CHA 25	0x2191	0x2190	0x219B	0x219A	0x2199	0x2198	0x2195	0x2194	0X2193	0x2192	0x219E
CHA 26	0x21A1	0x21A0	0x21AB	0x21AA	0x21A9	0x21A8	0x21A5	0x21A4	0x21A3	0x21A2	0x21AE
CHA 27	0x21B1	0x21B0	0x21BB	0x21BA	0x21B9	0x21B8	0x21B5	0x21B4	0x21B3	0x21B2	0x21BE
CHA 28	0x21C1	0x21C0	0x21CB	0x21CA	0x21C9	0x21C8	0x21C5	0x21C4	0x21C3	0x21C2	0x21CE
CHA 29	0x21D1	0x21D0	0x21DB	0x21DA	0x21D9	0x21D8	0x21D5	0x21D4	0x21D3	0x21D2	0x21DE
CHA 30	0x21E1	0x21E0	0x21EB	0x21EA	0x21E9	0x21E8	0x21E5	0x21E4	0x21E3	0x21E2	0x21EE
CHA 31	0x21F1	0x21F0	0x21FB	0x21FA	0x21F9	0x21F8	0x21F5	0x21F4	0x21F3	0x21F2	0x21FE
CHA 32	0x2201	0x2200	0x220B	0x220A	0x2209	0x2208	0x2205	0x2204	0x2203	0x2202	0x220E
CHA 33	0x2211	0x2210	0x221B	0x221A	0x2219	0x2218	0x2215	0x2214	0x2213	0x2212	0x221E
CHA 34	0x2221	0x2220	0x222B	0x222A	0x2229	0x2228	0x2225	0x2224	0x2223	0x2222	0x222E
CHA 35	0x2231	0x2230	0x223B	0x223A	0x2239	0x2238	0x2235	0x2234	0x2233	0x2232	0x223E
CHA 36	0x2241	0x2240	0x224B	0x224A	0x2249	0x2248	0x2245	0x2244	0x2243	0x2242	0x224E
CHA 37	0x2251	0x2250	0x225B	0x225A	0x2259	0x2258	0x2255	0x2254	0x2253	0x2252	0x225E
CHA 38	0x2261	0x2260	0x226B	0x226A	0x2269	0x2268	0x2265	0x2264	0x2263	0x2262	0x226E
CHA 39	0x2271	0x2270	0x227B	0x227A	0x2279	0x2278	0x2275	0x2274	0x2273	0x2272	0x227E
CHA 40	0x2281	0x2280	0x228B	0x228A	0x2289	0x2288	0x2285	0x2284	0x2283	0x2282	0x228E
CHA 41	0x2291	0x2290	0x229B	0x229A	0x2299	0x2298	0x2295	0x2294	0x2293	0x2292	0x229E
CHA 42	0x22A1	0x22A0	0x22AB	0x22AA	0x22A9	0x22A8	0x22A5	0x22A4	0x22A3	0x22A2	0x22AE
CHA 43	0x22B1	0x22B0	0x22BB	0x22BA	0x22B9	0x22B8	0x22B5	0x22B4	0x22B3	0x22B2	0x22BE
CHA 44	0x22C1	0x22C0	0x22CB	0x22CA	0x22C9	0x22C8	0x22C5	0x22C4	0x22C3	0x22C2	0x22CE
CHA 45	0x22D1	0x22D0	0x22DB	0x22DA	0x22D9	0x22D8	0x22D5	0x22D4	0x22D3	0x22D2	0x22DE
CHA 46	0x22E1	0x22E0	0x22EB	0x22EA	0x22E9	0x22E8	0x22E5	0x22E4	0x22E3	0x22E2	0x22EE
CHA 47	0x22F1	0x22F0	0x22FB	0x22FA	0x22F9	0x22F8	0x22F5	0x22F4	0x22F3	0x22F2	0x22FE
CHA 48	0x2301	0x2300	0x230B	0x230A	0x2309	0x2308	0x2305	0x2304	0x2303	0x2302	0x230E
CHA 49	0x2311	0x2310	0x231B	0x231A	0x2319	0x2318	0x2315	0x2314	0x2313	0x2312	0x231E
CHA 50	0x2321	0x2320	0x232B	0x232A	0x2329	0x2328	0x2325	0x2324	0x2323	0x2322	0x232E
CHA 51	0x2331	0x2330	0x233B	0x233A	0x2339	0x2338	0x2335	0x2334	0x2333	0x2332	0x233E
CHA 52	0x2341	0x2340	0x234B	0x234A	0x2349	0x2348	0x2345	0x2344	0x2343	0x2342	0x234E
CHA 53	0x2351	0x2350	0x235B	0x235A	0x2359	0x2358	0x2355	0x2354	0x2353	0x2352	0x235E
CHA 54	0x2361	0x2360	0x236B	0x236A	0x2369	0x2368	0x2365	0x2364	0x2363	0x2362	0x236E
CHA 55	0x2371	0x2370	0x237B	0x237A	0x2379	0x2378	0x2375	0x2374	0x2373	0x2372	0x237E
CHA 56	0x2381	0x2380	0x238B	0x238A	0x2389	0x2388	0x2385	0x2384	0x2383	0x2382	0x238E

Table 1-9. Uncore Performance Monitoring Registers (MSR) (Sheet 3 of 4)

Unit	Unit Status	Unit Ctrl	Ctr3	Ctr2	Ctr1	Ctr0	Ctrl3	Ctrl2	Ctrl1	Ctrl0	Extra
CHA 57	0x2391	0x2390	0x239B	0x239A	0x2399	0x2398	0x2395	0x2394	0x2393	0x2392	0x239E
CHA 58	0x23A1	0x23A0	0x23AB	0x23AA	0x23A9	0x23A8	0x23A5	0x23A4	0x23A3	0x23A2	0x23AE
CHA 59	0x23B1	0x23B0	0x23BB	0x23BA	0x23B9	0x23B8	0x23B5	0x23B4	0x23B3	0x23B2	0x23BE
CHA 60	0x23C1	0x23C0	0x23CB	0x23CA	0x23C9	0x23C8	0x23C5	0x23C4	0x23C3	0x23C2	0x23CE
CHA 61	0x23D1	0x23D0	0x23DB	0x23DA	0x23D9	0x23D8	0x23D5	0x23D4	0x23D3	0x23D2	0x23DE
CHA 62	0x23E1	0x23E0	0x23EB	0x23EA	0x23E9	0x23E8	0x23E5	0x23E4	0x23E3	0x23E2	0x23EE
CHA 63	0x23F1	0x23F0	0x23FB	0x23FA	0x23F9	0x23F8	0x23F5	0x23F4	0x23F3	0x23F2	0x23FE
MDF Blocks											
MDF 0	0x2801	0x2800	0x280B	0x280A	0x2809	0x2808	0x2805	0x2804	0x2803	0x2802	
MDF 1	0x2811	0x2810	0x281B	0x281A	0x2819	0x2818	0x2815	0x2814	0x2813	0x2812	
MDF 2	0x2821	0x2820	0x282B	0x282A	0x2829	0x2828	0x2825	0x2824	0x2823	0x2822	
MDF 3	0x2831	0x2830	0x283B	0x283A	0x2839	0x2838	0x2835	0x2834	0x2833	0x2832	
MDF 4	0x28A1	0x28A0	0x28AB	0x28AA	0x28A9	0x28A8	0x28A5	0x28A4	0x28A3	0x28A2	
MDF 5	0x28B1	0x28B0	0x28BB	0x28BA	0x28B9	0x28B8	0x28B5	0x28B4	0x28B3	0x28B2	
MDF 6	0x28C1	0x28C0	0x28CB	0x28CA	0x28C9	0x28C8	0x28C5	0x28C4	0x28C3	0x28C2	
MDF 7	0x28D1	0x28D0	0x28DB	0x28DA	0x28D9	0x28D8	0x28D5	0x28D4	0x28D3	0x28D2	
MDF 8	0x2941	0x2940	0x294B	0x294A	0x2949	0x2948	0x2945	0x2944	0x2943	0x2942	
MDF 9	0x2951	0x2950	0x295B	0x295A	0x2959	0x2958	0x2955	0x2954	0x2953	0x2952	
MDF 10	0x2961	0x2960	0x296B	0x296A	0x2969	0x2968	0x2965	0x2964	0x2963	0x2962	
MDF 11	0x2971	0x2970	0x297B	0x297A	0x2979	0x2978	0x2975	0x2974	0x2973	0x2972	
MDF 12	0x29E1	0x29E0	0x29EB	0x29EA	0x29E9	0x29E8	0x29E5	0x29E4	0x29E3	0x29E2	
MDF 13	0x29F1	0x29F0	0x29FB	0x29FA	0x29F9	0x29F8	0x29F5	0x29F4	0x29F3	0x29F2	
M2IOSF Blocks											
M2PCIe*											
M2IOSF 0	0x3201	0x3200	0x320B	0x320A	0x3209	0x3208	0x3205	0x3204	0x3203	0x3202	
M2IOSF 1	0x3211	0x3210	0x321B	0x321A	0x3219	0x3218	0x3215	0x3214	0x3213	0x3212	
M2IOSF 2	0x3221	0x3220	0x322B	0x322A	0x3229	0x3228	0x3225	0x3224	0x3223	0x3222	
M2IOSF 3	0x3231	0x3230	0x323B	0x323A	0x3239	0x3238	0x3235	0x3234	0x3233	0x3232	
M2IOSF 4	0x3241	0x3240	0x324B	0x324A	0x3249	0x3248	0x3245	0x3244	0x3243	0x3242	
M2IOSF 5	0x3251	0x3250	0x325B	0x325A	0x3259	0x3258	0x3255	0x3254	0x3253	0x3252	
M2IOSF 6	0x3261	0x3260	0x326B	0x326A	0x3269	0x3268	0x3265	0x3264	0x3263	0x3262	
M2IOSF 7	0x3271	0x3270	0x327B	0x327A	0x3279	0x3278	0x3275	0x3274	0x3273	0x3272	
M2IOSF 8	0x3281	0x3280	0x328B	0x328A	0x3289	0x3288	0x3285	0x3284	0x3283	0x3282	
M2IOSF 9	0x3291	0x3290	0x329B	0x329A	0x3299	0x3298	0x3295	0x3294	0x3293	0x3292	
IRP											IIO Clock
M2IOSF 0	0x3401	0x3400			0x3409	0x3408			0x3403	0x3402	0x340E

Table 1-9. Uncore Performance Monitoring Registers (MSR) (Sheet 4 of 4)

Unit	Unit Status	Unit Ctrl	Ctr3	Ctr2	Ctr1	Ctr0	Ctrl3	Ctrl2	Ctrl1	Ctrl0	Extra
M2IOSF 1	0x3411	0x3410			0x3419	0x3418			0x3413	0x3412	0x341E
M2IOSF 2	0x3421	0x3420			0x3429	0x3428			0x3423	0x3422	0x342E
M2IOSF 3	0x3431	0x3430			0x3439	0x3438			0x3433	0x3432	0x343E
M2IOSF 4	0x3441	0x3440			0x3449	0x3448			0x3443	0x3442	0x344E
M2IOSF 5	0x3451	0x3450			0x3459	0x3458			0x3453	0x3452	0x345E
M2IOSF 6	0x3461	0x3460			0x3469	0x3468			0x3463	0x3462	0x346E
M2IOSF 7	0x3471	0x3470			0x3479	0x3478			0x3473	0x3472	0x347E
M2IOSF 8	0x3481	0x3480			0x3489	0x3488			0x3483	0x3482	0x348E
M2IOSF 9	0x3491	0x3490			0x3499	0x3498			0x3493	0x3492	0x349E
TC											
M2IOSF 0	0x3001	0x3000	0x300B	0x300A	0x3009	0x3008	0x3005	0x3004	0x3003	0x3002	
M2IOSF 1	0x3011	0x3010	0x301B	0x301A	0x3019	0x3018	0x3015	0x3014	0x3013	0x3012	
M2IOSF 2	0x3021	0x3020	0x302B	0x302A	0x3029	0x3028	0x3025	0x3024	0x3023	0x3022	
M2IOSF 3	0x3031	0x3030	0x303B	0x303A	0x3039	0x3038	0x3035	0x3034	0x3033	0x3032	
M2IOSF 4	0x3041	0x3040	0x304B	0x304A	0x3049	0x3048	0x3045	0x3044	0x3043	0x3042	
M2IOSF 5	0x3051	0x3050	0x305B	0x305A	0x3059	0x3058	0x3055	0x3054	0x3053	0x3052	
M2IOSF 6	0x3061	0x3060	0x306B	0x306A	0x3069	0x3068	0x3065	0x3064	0x3063	0x3062	
M2IOSF 7	0x3071	0x3070	0x307B	0x307A	0x3079	0x3078	0x3075	0x3074	0x3073	0x3072	
M2IOSF 8	0x3081	0x3080	0x308B	0x308A	0x3089	0x3088	0x3085	0x3084	0x3083	0x3082	
M2IOSF 9	0x3091	0x3090	0x309B	0x309A	0x3099	0x3098	0x3095	0x3094	0x3093	0x3092	
PCU											Filter
PCU	0x2FC1	0x2FC0	0x2FCB	0x2FCA	0x2FC9	0x2FC8	0x2FC5	0x2FC4	0x2FC3	0x2FC2	0x2FCE

There are a number of free-running counters in each IIO Stack that collect counts for Input/Output Bandwidth for each Port. The MSR addresses used to access that state are detailed in the following tables.

Table 1-10. Free-running IIO Bandwidth In Counters in MSR space

	Port 7 BW In	Port 6 BW In	Port 5 BW In	Port 4 BW In	Port 3 BW In	Port 2 BW In	Port 1 BW In	Port 0 BW In
M2IOSF 0	0x3807	0x3806	0x3805	0x3804	0x3803	0x3802	0x3801	0x3800
M2IOSF 1	0x3817	0x3816	0x3815	0x3814	0x3813	0x3812	0x3811	0x3810
M2IOSF 2	0x3827	0x3826	0x3825	0x3824	0x3823	0x3822	0x3821	0x3820
M2IOSF 3	0x3837	0x3836	0x3835	0x3834	0x3833	0x3832	0x3831	0x3830
M2IOSF 4	0x3847	0x3846	0x3845	0x3844	0x3843	0x3842	0x3841	0x3840
M2IOSF 5	0x3857	0x3856	0x3855	0x3854	0x3853	0x3852	0x3851	0x3850
M2IOSF 6	0x3867	0x3866	0x3865	0x3864	0x3863	0x3862	0x3861	0x3860

Table 1-10. Free-running IIO Bandwidth In Counters in MSR space

	Port 7 BW In	Port 6 BW In	Port 5 BW In	Port 4 BW In	Port 3 BW In	Port 2 BW In	Port 1 BW In	Port 0 BW In
M2IOSF 7	0x3877	0x3876	0x3875	0x3874	0x3873	0x3872	0x3871	0x3870
M2IOSF 8	0x3887	0x3886	0x3885	0x3884	0x3883	0x3882	0x3881	0x3880
M2IOSF 9	0x3897	0x3896	0x3895	0x3894	0x3893	0x3892	0x3891	0x3890

Table 1-11. Free-running IIO Bandwidth Out Counters in MSR space

	Port 7 BW Out	Port 6 BW In	Port 5 BW In	Port 4 BW In	Port 3 BW In	Port 2 BW In	Port 1 BW In	Port 0 BW In
M2IOSF 0	0x380F	0x380E	0x380D	0x380C	0x380B	0x380A	0x3809	0x3808
M2IOSF 1	0x381F	0x381E	0x381D	0x381C	0x381B	0x381A	0x3819	0x3818
M2IOSF 2	0x382F	0x382E	0x382D	0x382C	0x382B	0x382A	0x3829	0x3828
M2IOSF 3	0x383F	0x383E	0x383D	0x383C	0x383B	0x383A	0x3839	0x3838
M2IOSF 4	0x384F	0x384E	0x384D	0x384C	0x384B	0x384A	0x3849	0x3848
M2IOSF 5	0x385F	0x385E	0x385D	0x385C	0x385B	0x385A	0x3859	0x3858
M2IOSF 6	0x386F	0x386E	0x386D	0x386C	0x386B	0x386A	0x3869	0x3868
M2IOSF 7	0x387F	0x387E	0x387D	0x387C	0x387B	0x387A	0x3879	0x3878
M2IOSF 8	0x388F	0x388E	0x388D	0x388C	0x388B	0x388A	0x3889	0x3888
M2IOSF 9	0x389F	0x389E	0x389D	0x389C	0x389B	0x389A	0x3899	0x3898

Note: Refer to each unit's performance monitoring section for any related state not covered here.

1.8.2 Uncore Performance Monitoring State in PCICFG space

The addresses for all basic PMON state addressed through PCICFG space are laid out in the following table. Each such block will have a PCICFG B:D:F and DeviceID. The registers are presented as offsets to the PMON block's base address.

Table 1-12. Uncore Performance Monitoring Registers (PCICFG) (Sheet 1 of 2)

Unit	Unit Status	Unit Ctrl	Ctr3	Ctr2	Ctr1	Ctr0	Ctrl3	Ctrl2	Ctrl1	Ctrl0	Extra	
Unit - PMON Block			PCICFG B:D:F Base Address			Device ID						
								0X3251				
IMC0			MEM0_BAR, B30:D0:F1,0xD8									
IMC1			MEM1_BAR, B30:D0:F1,0xDC									
IMC2			MEM2_BAR, B30:D0:F1,0xE0									
IMC3			MEM3_BAR, B30:D0:F1,0xE4								DCLK Ctrl	DCLK Ctr

Table 1-12. Uncore Performance Monitoring Registers (PCICFG) (Sheet 2 of 2)

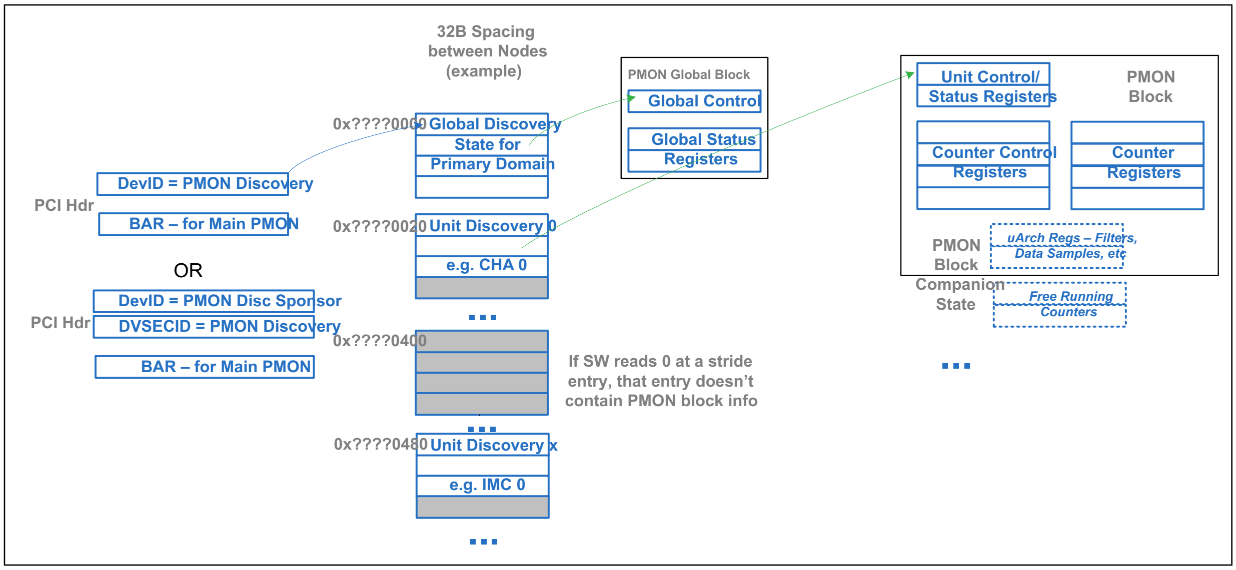
Unit	Unit Status	Unit Ctrl	Ctr3	Ctr2	Ctr1	Ctr0	Ctrl3	Ctrl2	Ctrl1	Ctrl0	Extra	
Channel 0	0x2285C	0x22800	0x22820	0x22818	0x22810	0x22808	0x2284c	0x22848	0x22844	0x22840	0x22 838	0x22 854
Channel 1	0x2A85C	0x2A800	0x2A820	0x2A818	0x2A810	0x2A808	0x2A84c	0x2A848	0x2A844	0x2A840	0x2A 838	0x2A 854
iMC Free Running Counters							DCLK	rpq_active_cycles	wpq_active_cycles			
							0x22B0	0x2318	0x2320			
M2M - for iMC0			B30:D12:F0			0x324A						
M2M - for iMC1			B30:D13:F0			0x324A						
M2M - for iMC2			B30:D14:F0			0x324A						
M2M - for iMC3			B30:D15:F0			0x324A					See M2M Section	
M2M	0x4A8	0x438	0x458	0x450	0x448	0x440	0x480	0x478	0x470	0x468		
M2UPI - Link 0			B30:D5:F1			0x3246						
M2UPI - Link 1			B30:D6:F1			0x3246						
M2UPI - Link 2			B30:D7:F1			0x3246						
M2UPI - Link 3			B30:D8:F1			0x3246						
M2UPI	0xF8	0xA0	0xC0	0xB8	0xB0	0xA8	0xF0	0xE8	0xE0	0xD8		
UPI LL - Link 0			B30:D1:F1			0x3241						
UPI LL - Link 1			B30:D2:F1			0x3241						
UPI LL - Link 2			B30:D3:F1			0x3241						
UPI LL - Link 3			B30:D4:F1			0x3241						
UPI LL	0x38C	0x318	0x338	0x330	0x328	0x320	0x368	0x360	0x358	0x350		

1.9 Intro to Discovery - Self Describing HW

In the 5th Gen Intel® Xeon® Scalable Processor, the self describing HW starts by reading through an MMIO page worth of information, SW can 'discover' all the standard PMON registers in the global block followed by all the standard PMON registers in each of the units.

Non-standard PMON registers will not be included. For example, free running counters like the Mem/IIO BW counters, fixed counters like DCLK and extra filtering/matching registers such as the ones in the CHA and M2M. SW tools that support these UARCH specific extensions to the standard monitoring capabilities, will have to hard code access as they had before.

Figure 1-9. Discovery - An Overview

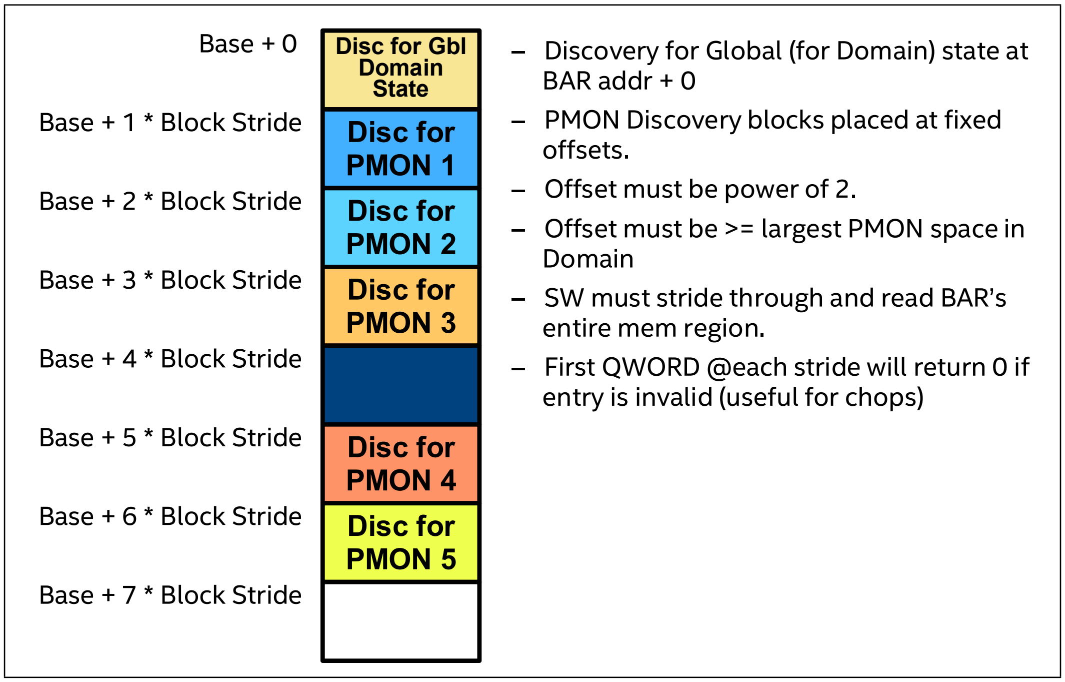


The first step is for the SW to find the device header sponsoring the MMIO page worth of PMON discovery. To do this, SW, while walking through the list of available PCI headers, should look for either the PCI header labeled PMON discovery or the DVSEC substructure (that is, extended capability) labeled PMON discovery. Once found, the SW can read the Base Address Register (BAR) and page size to determine the bounds of the discovery information.

Note: Example code has been provided in Section 1.10.1.

The following diagram illustrates the basic structure of PMON discovery within the page. SW should first read the Global Discovery information (see Section 1-11, "Discovery - Global State") from offset 0x0.

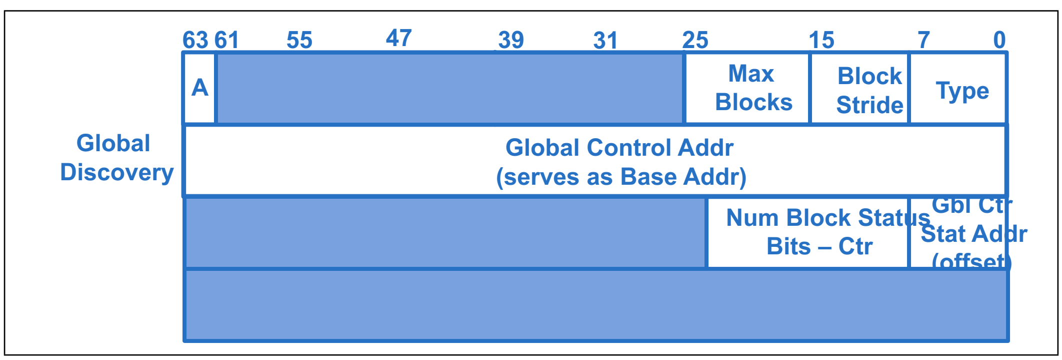
Figure 1-10. Discovery - Visual Guide for How SW Strides Page



In reading the Global discovery, SW can determine where the global control/status registers are, how large each block of Unit Discovery information is (Block Stride) and the number of strides it will take to reach the end of the page (Max Blocks).

1.9.1 Global Discovery

Figure 1-11. Discovery - Global State



The global entries in the PMON Discovery page are to inform SW:

- How to address the global control (Global Control Addr).
- How to address the other registers that form the global block.
- What address space these registers are accessed through.
- How to read through the rest of the Discovery page to find all the Unit discovery.

Table 1-13. Global Discovery– Field Definitions

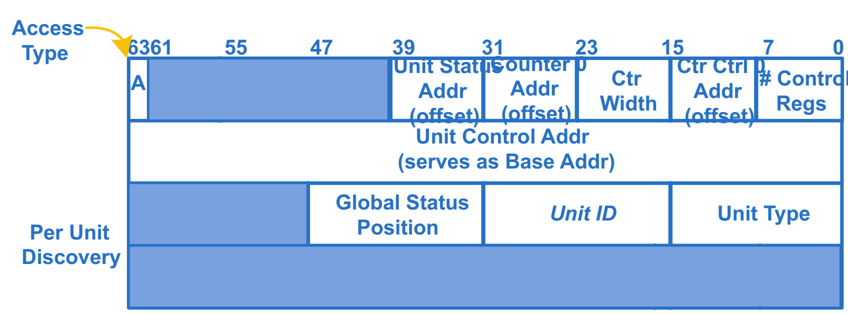
	Field	Bits	Description
Global Node +0	Access Type	63:61	Global State is accessed through 00 - MSR space 01 - MMIO space 10 - PCICFG space
	rsv	60:26	Reserved
	Max Blocks	25:16	The number of strides (0 for the global discovery node) the SW will need to make through the address space to ensure that all the unit discovery state from the domain has been found.
	Block Stride	15:8	The length of each stride represents the amount of space reserved for each block of discovery. From the base address, SW will need to stride through MaxBlocks-1 times from the base address to identify all discovery state.
	Type	7:0	Domain Type
Global Node +1	Global Control Address	63:0	Address to the Global Control Register
Global Node +2	rsv	63:24	Reserved.
	Num Block Stat Addr Bits - Counters	23:8	How many status bits are allocated to track overflows? For cases there are more than 64 status bits, SW should divide this value by 64 to calculate the number of contiguously addressed counter status registers.
	Gbl Ctr Stat Addr (offset)	7:0	8b offset from Global Control Address to first counter status register.

SW should then stride the rest of the MMIO page to identify each PMON block. Any non-0 entry provides discovery information about a unit's PMON block.

```
For i = 0; i <= MaxBlocks - 1; i += BlockStride; {
    if page_of_discovery[i] != 0 process_unit_discovery()_
```

1.9.2 Unit Discovery

Figure 1-12. Discovery - Unit State



Each of the blocks of unit discovery information tells the SW:

- The address space these registers are accessed through.
- How to address the global control (Unit Control Addr).

- Given the unit control's address, how to address the other standard registers in each PMON block - includes counter control/counter pairs, the unit control and unit status registers.
- The 'Unit ID' used to determine which of the 5th Gen Intel® Xeon® Scalable Processor's event files is associated with this PMON block.

Table 1-14. Unit Discovery– Field Definitions

	Field	Bits	Description
Global Node +0	Access Type	63:61	Unit State is accessed through 00 - MSR space 01 - MMIO space 10 - PCICFG space
	rsv	60:40	Reserved
	Unit Status Addr (offset)	39:32	8b offset from the unit control address to the first unit status register.
	Counter 0 Address (offset)	31:24	8b offset from the unit control address to the first counter register. Additional counters are contiguously spaced from the first.
	Counter Width	23:16	Number of bits in data register.
	Counter Control 0 Address (offset)	15:8	8b offset from Unit Control Address to first counter control register. Additional counter controls are contiguously spaced from first.
	Num Control Regs	7:0	Number of counter control registers paired with data registers in this Unit.
Global Node +1	Unit Control Address	63:0	Address to this unit's control register.
Global Node +2	rsv	63:48	Reserved.
	Global Status Position	47:32	16b field to tell SW which bit in the Global Status belongs to the PMON block.
	Unit ID	31:16	Which # of this unit's type? For cases where there are more than one instance of a particular unit, this ID's the specific PMON block for the Unit Type. For example, CHA #4 or iMC PMON block #2.
	Unit Type	15:0	What kind of Unit is the PMON block associated with? Each Unit of a Unit Type will offer the same event list. Each Unit of a Unit Type will offer the same UARCH specific PMON HW.

1.10 Some Guidance for SW

1.10.1 On Finding PMON Discovery and Reading It

The following code details how to find the device sponsoring PMON discovery, how to address the MMIO page worth of discovery, how traverse through it, and how to find all the PMON registers.

```
/* Capability ID for discovery table device */
#define UNCORE_EXT_CAP_ID_DISCOVERY                    0x23
/* DVSEC offset */
#define UNCORE_DISCOVERY_DVSEC_OFFSET                  0x8
/* mask of DVSEC_ID */
#define UNCORE_DISCOVERY_DVSEC_ID_MASK                 0xffff
/* PMON discovery entry type ID */
```

```
#define UNCORE_DISCOVERY_DVSEC_ID_PMON              0x1
/* mask of BIR */
#define UNCORE_DISCOVERY_DVSEC_BIR_MASK             0x7
/* discovery table size */
#define UNCORE_DISCOVERY_MAP_SIZE                   0x80000

struct uncore_global_discovery {
	union {
		u64	table1;
		struct {
			u64	type : 8,
				stride : 8,
				max_units : 10,
				__reserved_1 : 36,
				access_type : 2;
		};
	};
	union {
		u64	table2;
		u64	global_ctl;
	};
	union {
		u64	table3;
		struct {
		u64	status_offset : 8,
			num_status : 16,
			__reserved_2 : 40;
		};
	};
};

struct uncore_unit_discovery {
	union {
		u64	table1;
		struct {
		u64	num_regs : 8,
			ctl_offset : 8,
			bit_width : 8,
			ctr_offset : 8,
			status_offset : 8,
			__reserved_1 : 22,
			access_type : 2;
		};
	};
	union {
		u64	table2;
		u64	box_ctl;
	};
	union {
```

```
            u64    table3;
            struct {
                   u64    box_type : 16,
                          box_id : 16,
                          __reserved_2 : 32;
            };
      };
};


/* Go through the entire PCI devices tree */
while ((dev = pci_get_device(PCI_VENDOR_ID_INTEL, PCI_ANY_ID, dev)) !=
NULL) {

      /* Walk the Extended Capability structures looking for a DVSEC
structure with unique capability ID 0x23 */
      while ((dvsec = pci_find_next_ext_capability(dev, dvsec,
UNCORE_EXT_CAP_ID_DISCOVERY))) {

            /* read the DVSEC_ID (15:0) */
            pci_read_config_dword(dev, dvsec +
UNCORE_DISCOVERY_DVSEC_OFFSET, &val);
            entry_id = val & UNCORE_DISCOVERY_DVSEC_ID_MASK;

            /* check if it is PMON discovery entry */
            if (entry_id == UNCORE_DISCOVERY_DVSEC_ID_PMON) {

                  /* read BIR value (2:0) */
                  pci_read_config_dword(dev, dvsec +
UNCORE_DISCOVERY_DVSEC_OFFSET + 4, &bir);
                  bir = bir & UNCORE_DISCOVERY_DVSEC_BIR_MASK;

                  /* calculate the BAR offset of global discovery table
*/
                  bar_offset = 0x10 + (bir * 4);

                  /* read the BAR address of global discovery table */
                  pci_read_config_dword(dev, bar_offset, &pci_dword);

                  /* Map whole discovery table */
                  addr = pci_dword & ~(PAGE_SIZE - 1);
                  io_addr = ioremap(addr, UNCORE_DISCOVERY_MAP_SIZE);

                  /* Read Global Discovery table */
                  memcpy_fromio(&global, io_addr, sizeof(struct
uncore_global_discovery));

                  /* Read Unit Discovery table one by one */
                  for (i = 0; i < global.max_units; i++) {
```

```
                                memcpy_fromio(&unit, io_addr + (i + 1) *
(global.stride * 8), sizeof(struct uncore_unit_discovery));

                                /* parse the unit discovery table  here*/
                        }
                }
        }

}
```

1.10.2 On Finding the Package's Bus Number for Uncore PMON Registers in PCICFG Space

PCI-based uncore units in 5th Gen Intel® Xeon® Scalable Processor can be found using bus, device and functions numbers. However, the **bus number** (*busno*) has to be found dynamically in each package. The code is embedded next.

First, for each package, it is necessary to read the node ID offset in the Ubox. That needs to match the GID offset of the Ubox in a specific pattern to get the bus no for the package. This *busno* can then be used with the given Device : Function (D:F) listed with each box's counters that are accessed through PCICfg space (Table 1.8.2, "Uncore Performance Monitoring State in PCICFG space").

Note: The one undefined piece in the following code is PCI_Read_Ulong, a function that simply reads the value from the PCI address. This function, or ones like it, can be found in a more general PCI library the composition of which is OS dependent.

Unfortunately, a link to a suitable version of the library was not readily available. Next are links to a comparable open source version of the library. Included for reference:

https://github.com/opcm/pcm/blob/master/ pci.h and pci.cpp

```
#define DRV_IS_PCI_VENDOR_ID_INTEL             0x8086
#define VENDOR_ID_MASK                         0x0000FFFF
#define DEVICE_ID_MASK                         0xFFFF00000
#define DEVICE_ID_BITSHIFT                     16

#define PCI_ENABLE                             0x80000000
#define FORM_PCI_ADDR(bus,dev,fun,off)         (((PCI_ENABLE))      | \
                                                ((bus & 0xFF) << 16)| \
                                                ((dev & 0x1F) << 11)| \
                                                ((fun & 0x07) << 8) | \
                                                ((off & 0xFF) << 0))


#define EMR_SERVER_SOCKETID_UBOX_DID   0x3250

//the below LNID and GID applies to Emerald Rapids Server
#define UNC_SOCKETID_UBOX_LNID_OFFSET   0xC0
```

```
#define UNC_SOCKETID_UBOX_GID_OFFSET     0xD4

for (bus_no = 0; bus_no < 256; bus_no++) {
  for (device_no = 0; device_no < 32; device_no++) {
   for (function_no = 0; function_no < 8; function_no++) {

     // find bus, device, and function number for socket ID UBOX device
     pci_address = FORM_PCI_ADDR(bus_no, device_no, function_no, 0);
     value = PCI_Read_Ulong(pci_address);

     vendor_id = value & VENDOR_ID_MASK;
     device_id = (value & DEVICE_ID_MASK) >> DEVICE_ID_BITSHIFT;

     if (vendor_id != DRV_IS_PCI_VENDOR_ID_INTEL) {
       continue;
        }
     if (device_id == EMR_SERVER_SOCKETID_UBOX_DID) {
      // first get node id for the local socket
      pci_address = FORM_PCI_ADDR(bus_no, device_no, function_no,
                    UNC_SOCKETID_UBOX_LNID_OFFSET);
      gid = PCI_Read_Ulong(pci_address) & 0x00000007;

      // Get the node id mapping register:
      // Basic idea is to read the Node ID Mapping Register (below)
      // and match one of the nodes with gid that we read above
      // from the Node ID configuration register (above).
      // Every three bits in the Node ID Mapping Register maps to a
      // particular node (or package). Bits 2:0 maps to package 0,
      // bits 5:3 maps to package 1, and so on. Thus, we have to
      // parse every triplet of bits to find the match.

      pci_address = FORM_PCI_ADDR(bus_no, device_no, function_no,
                    UNC_SOCKETID_UBOX_GID_OFFSET);
      mapping = PCI_Read_Ulong(pci_address);


       for (i = 0; i < 8; i++){
        if (nodeid == ((mapping >> (3 * i)) & 0x7)) {
              gid = i;
              break;
        }
       }

       UNC_UBOX_package_to_bus_map[gid] = bus_no;
       }
   }
  }

}
```

1.10.3 On Resolving Addresses for Uncore PMON Registers in MMIO Space

MMIO-based uncore units in the 5th Gen Intel® Xeon® Scalable Processor can be found by taking the DeviceID and looking up the Base Address Offset (BAR) that governs that unit's registers. For the 5th Gen Intel® Xeon® Scalable Processor, the BAR lookup is a two-step process as outlined next.

Once the base address has been resolved, simply add the published offsets to reference the PMON registers (Table 1.8.2, "Uncore Performance Monitoring State in PCICFG space").

```
/* MMIO_BASE found at Bus U0, Device 0, Function 1, offset D0h. */
#define EMR_X_IMC_MMIO_BASE_OFFSET  0xd0
#define EMR_X_IMC_MMIO_BASE_MASK    0x1FFFFFFF
/* MEM0_BAR found at Bus U0, Device 0, Function 1, offset D8h. */
#define EMR_X_IMC_MMIO_MEM0_OFFSET  0xd8
#define EMR_X_IMC_MMIO_MEM_STRIDE   0x04
#define EMR_X_IMC_MMIO_MEM_MASK     0x7FF
/*
* Each IMC has two channels.
* The offset starts from 0x22800 with stride 0x8000
*/
#define EMR_IMC_MMIO_CHN_OFFSET     0x22800
#define EMR_IMC_MMIO_CHN_STRIDE     0x8000
/* IMC MMIO size*/
#define EMR_X_IMC_MMIO_SIZE         0x4000


/*
* pkg_id: Socket id
* imc_idx: The IMC index
* channel_idx: The channel index
*/
Void *map_imc_pmon(int pkg_id, int imc_idx, int channel_idx)
{
       struct pci_dev *pdev = NULL;
       resource_size_t addr;
       u32 pci_dword;
       void *io_addr;
       int mem_offset;

/*
       * Device ID of Bus U0, Device 0, Function 1 is 0x3251 */
       * Get its pdev on the specific socket.
*/
       while(1){
              pdev = pci_get_device(PCI_VENDOR_ID_INTEL, 0x3251, pdev);
```

```
                              if ((!pdev) || (pdev->bus ==
UNC_UBOX_package_to_bus_map[pkg_id]))
                                             break;
              }
       if (!pdev)
              return NULL;

       /* read MEMn addr (51:23) from MMIO_BASE register */
       pci_read_config_dword(pdev, EMR_IMC_MMIO_BASE_OFFSET, &pci_dword);
       addr = (pci_dword & EMR_IMC_MMIO_BASE_MASK) << 23;

       /* read MEMn addr (22:12) from MEMn_BAR register */
        mem_offset = EMR_IMC_MMIO_MEM0_OFFSET + mem_idx *
EMR_IMC_MMIO_MEM_STRIDE;
       pci_read_config_dword(pdev, mem_offset, &pci_dword);
       addr |= (pci_dword & EMR_IMC_MMIO_MEM_MASK) << 12;

       /* IMC PMON registers start from PMONUNITCTRL */
       addr += EMR_IMC_MMIO_CHN_OFFSET + channel_idx *
EMR_IMC_MMIO_CHN_STRIDE;

       /* map the IMC PMON registers */
       io_addr = ioremap(addr, EMR_IMC_MMIO_SIZE);

       return io_addr;
}
```

1.10.4 Setting up a Monitoring Session

On a HW reset all the counters are disabled. Enabling is hierarchical. So the following steps, which include programming the event control registers and enabling the counters to begin collecting events, must be taken to set up a monitoring session. Section 1.10.5, “Reading the Sample Interval” covers the steps to stop/re-start counter registers during a monitoring session.

Global Settings in the UBox:

a) Freeze all the uncore counters by setting U_MSR_PMON_GLOBAL_CTL.frz_all to 1.

Note: Necessary for Ubox monitoring.

OR (if box level freeze control preferred)

a) Freeze the box’s counters while setting up the monitoring session.

For example, set Cn_MSR_PMON_BOX_CTL.frz to 1.

b) Select event to monitor if the event control register hasn’t been programmed:

Program the *.ev_sel* and *.umask* bits in the control register with the encoding necessary to capture the requested event along with any signal conditioning bits (*.thresh*/*.edge_det*/*.invert*) used to qualify the event.

Back to the box level:

c) Reset counters in each box to ensure no stale values have been acquired from previous sessions. Resetting the control registers, particularly those that will not be used is also recommended if for no other reason than to prevent errant overflows. To reset both the counters and control registers write the following registers:
   - For each CHAx, set Cn_MSR_PMON_UNIT_CTL[9:8] to 0x300.
   - For each MDFx, set Cn_MSR_PMON_UNIT_CTL[9:8] to 0x300.
   - For each DRAM Channel, set MCn_CHy_MMIO_PMON_UNIT_CTL[9:8] to 0x300
   - For each M2M, set M2M_PCI_PMON_UNIT_CTL[9:8] to 0x300
   - Set PCU_MSR_PMON_UNIT_CTL[9:8] to 0x300.
   - For each Intel® UPI link, set M2_Ly_PCI_PMON_UNIT_CTL[9:8] to 0x300.
   - For each Intel® UPI link, set UPI_Ly_PCI_PMON_UNIT_CTL[9:8] to 0x300.
   - For each IIO stack, set M2n_PCI_PMON_UNIT_CTL[9:8] to 0x300.
   - For each IIO stack, set IIOn_MSR_PMON_UNIT_CTL[9:8] to 0x300.
   - For each CXL stack, set CXLCM_MMIO_PMON_UNIT_CTL[9:8] to 0x300.
   - For each CXL stack, set CXLDP_MMIO_PMON_UNIT_CTL[9:8] to 0x300.
   - For each IIO stack, set IRPn_MSR_PMON_UNIT_CTL[9:8] to 0x300.

d) Select how to gather data. *If polling, skip to e*. If sampling:
   To set up a **sample interval**, the SW can pre-program the data register with a value of [2^(register bit width - up to 48) - sample interval length]. Doing so allows the SW, 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’ is a common use of this mechanism.

   That is, to stop counting and receive notification when the 1,000,000th data flit is transmitted from Intel UPI on Link 0.
   - Set UPI_L0_PCI_PMON_CTR1 to (2^48- 1000)
   - Set UPI_L0_PCI_PMON_CTL1.ev_sel to 0x2
   - Set UPI_L0_PCI_PMON_CTL1.umask to 0xF
   - Set U_MSR_PMON_GLOBAL_CTL.pmi_core_sel to which core the monitoring thread is executing on.

e) Enable counting at the global level by setting the U_MSR_PMON_GLOBAL_CTL.frz bit to 0.

OR

e) Enable counting at the box level by unfreezing the counters in each box.
   For example, set Cn_MSR_PMON_BOX_CTL.frz to 0

   And with that, counting will begin.

1.10.5 Reading the Sample Interval

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

a) **Polling** - before reading, it is recommended that software freeze the counters at either the Global level (U_MSR_PMON_GLOBAL_CTL.frz_all) or in each box with active counters (by setting *_PMON_UNIT_CTL.frz to 1). After reading the event counts from the counter registers, the monitoring agent can choose to reset the event counts to avoid event-count wrap-around; or resume the counter register without resetting their values. The latter choice will require the monitoring agent to check and adjust for potential wrap-around situations.

b) **Frozen** counters - If software set the counters to freeze on overflow and send notification when it happens, the next question is: Who caused the freeze?

Overflow bits are stored hierarchically within the uncore. First, the SW should read the U_MSR_PMON_GLOBAL_STATUS.ov_* bits to determine which box(es) sent an overflow. Then read that box's *_PMON_GLOBAL_STATUS.ov field to find the overflowing counter.

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

Note: Certain boxes may have more than one PMON block (for example, IMC has a PMON block in each channel). It may be necessary to read all STATUS registers in the box to determine which counter overflowed.

1.10.6 Enabling a New Sample Interval from Frozen Counters

a) **Clear all uncore counters**: For each box in which counting occurred, set *_PMON_BOX_CTL.rst_ctrs to 1.

b) **Clear all overflow bits**. This includes clearing U_MSR_PMON_GLOBAL_STATUS.ov_* as well as any *_BOX_STATUS registers that have their overflow bits set.

For example, if counter 3 in Intel UPI Link 1 overflowed, software should set UPI_L1_PCI_PMON_BOX_STATUS.ov[3] to 1 to clear the overflow.

c) **Create the next sample**: Reinitialize the sample by setting the monitoring data register to (2^48 - sample_interval). Or set up a new sample interval as outlined in Section 1.10.4, "Setting up a Monitoring Session".

d) **Re-enable counting**: Set U_MSR_PMON_GLOBAL_CTL.frz_all to 0.

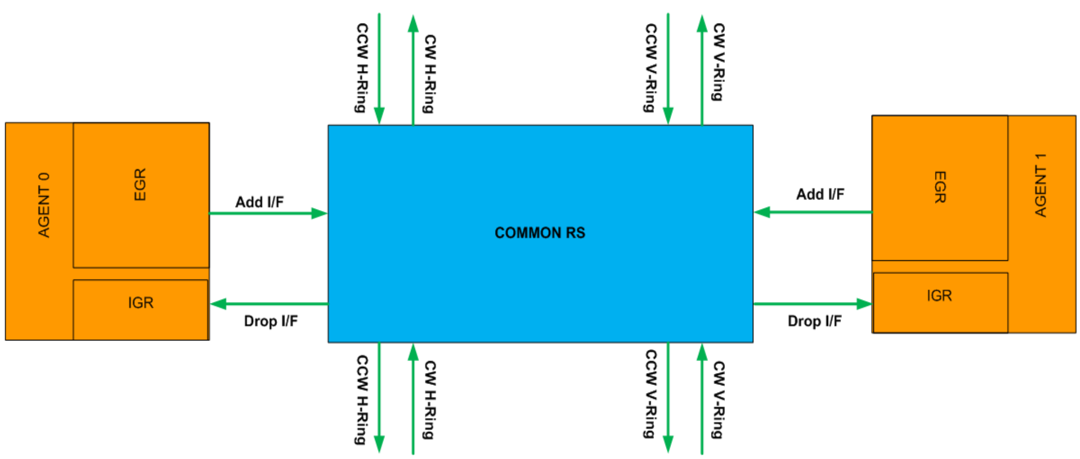
2 5th Gen Intel® Xeon® Scalable Processor Uncore Performance Monitoring

2.1 Mesh Performance Monitoring

For all boxes that must communicate with the mesh, there are a common set of events to capture various kinds of information about traffic flowing through their connection to the Mesh. The same encodings are used to request the mesh events in each box.

This common mesh stop event list is available in the CHA, M2PCIe* and M3 Intel® UPI.

Note: The common mesh stop events for M2M would have an additional bit enabled for its programming.The event list is available in Section 2.6.4, "Intel® UPI LL Box Events Ordered By Code".



2.1.1 Mesh Performance Monitoring Events

There are events to track information related to all traffic passing through each box's connection to the mesh.

Credit tracking and stalls due to lack of credits:

- Credits are required for each row (for example, transgress) destination through either side of the mesh stop for each path (Address [AD] and Block [BL]).
- Mesh stop events.
- To track the ingress or egress traffic, mesh utilization (broken down by direction and ring type), bypass statistics and more.

- Bounce and starvation events.
- Events to help recognize when the mesh is becoming or is saturated.

2.1.2 CMS Box Events Ordered By Code

The following table summarizes the directly measured CMS box events.

Table 2-1. Measured CMS Box Events (Sheet 1 of 3)

Symbol Name	Event Code	Ctrs	Description
AG0_AD_CRD_ACQUIRED0	0x80		CMS Agent0 AD Credits Acquired
AG0_AD_CRD_ACQUIRED1	0x81		CMS Agent0 AD Credits Acquired
AG0_AD_CRD_OCCUPANCY0	0x82		CMS Agent0 AD Credits Occupancy
AG0_AD_CRD_OCCUPANCY1	0x83		CMS Agent0 AD Credits Occupancy
AG0_BL_CRD_ACQUIRED0	0x88		CMS Agent0 BL Credits Acquired
AG0_BL_CRD_ACQUIRED1	0x89		CMS Agent0 BL Credits Acquired
AG0_BL_CRD_OCCUPANCY0	0x8A		CMS Agent0 BL Credits Occupancy
AG0_BL_CRD_OCCUPANCY1	0x8B		CMS Agent0 BL Credits Occupancy
AG1_AD_CRD_ACQUIRED0	0x84		CMS Agent1 AD Credits Acquired
AG1_AD_CRD_ACQUIRED1	0x85		CMS Agent1 AD Credits Acquired
AG1_AD_CRD_OCCUPANCY0	0x86		CMS Agent1 AD Credits Occupancy
AG1_AD_CRD_OCCUPANCY1	0x87		CMS Agent1 AD Credits Occupancy
AG1_BL_CRD_ACQUIRED0	0x8C		CMS Agent1 BL Credits Acquired
AG1_BL_CRD_ACQUIRED1	0x8D		CMS Agent1 BL Credits Acquired
AG1_BL_CRD_OCCUPANCY0	0x8E		CMS Agent1 BL Credits Occupancy
AG1_BL_CRD_OCCUPANCY1	0x8F		CMS Agent1 BL Credits Occupancy
CMS_CLOCKTICKS	0xC0		CMS Clockticks
DISTRESS_ASSERTED	0xAF		Distress signal asserted
EGRESS_ORDERING	0xBA		Egress Blocking due to Ordering requirements
HORZ_RING_AD_IN_USE	0xB6		Horizontal AD Ring In Use
HORZ_RING_AKC_IN_USE	0xBB		Horizontal AK Ring In Use
HORZ_RING_AK_IN_USE	0xB7		Horizontal AK Ring In Use
HORZ_RING_BL_IN_USE	0xB8		Horizontal BL Ring in Use
HORZ_RING_IV_IN_USE	0xB9		Horizontal IV Ring in Use
MISC_EXTERNAL	0xE6		Miscellaneous Events (mostly from MS2IDI)
RING_BOUNCES_HORZ	0xAC		Messages that bounced on the Horizontal Ring.
RING_BOUNCES_VERT	0xAA		Messages that bounced on the Vertical Ring.
RING_SINK_STARVED_HORZ	0xAD		Sink Starvation on Horizontal Ring
RING_SINK_STARVED_VERT	0xAB		Sink Starvation on Vertical Ring
RING_SRC_THRTL	0xAE		Source Throttle
RxR_BUSY_STARVED	0xE5		Transgress Injection Starvation
RxR_BYPASS	0xE2		Transgress Ingress Bypass

Table 2-1. Measured CMS Box Events (Sheet 2 of 3)

Symbol Name	Event Code	Ctrs	Description
RxR_CRD_STARVED	0xE3		Transgress Injection Starvation
RxR_CRD_STARVED_1	0xE4		Transgress Injection Starvation
RxR_INSERTS	0xE1		Transgress Ingress Allocations
RxR_OCCUPANCY	0xE0		Transgress Ingress Occupancy
STALL0_NO_TxR_HORZ_CRD_AD_AG0	0xD0		Stall on No AD Agent0 Transgress Credits
STALL0_NO_TxR_HORZ_CRD_AD_AG1	0xD2		Stall on No AD Agent1 Transgress Credits
STALL0_NO_TxR_HORZ_CRD_BL_AG0	0xD4		Stall on No BL Agent0 Transgress Credits
STALL0_NO_TxR_HORZ_CRD_BL_AG1	0xD6		Stall on No BL Agent1 Transgress Credits
STALL1_NO_TxR_HORZ_CRD_AD_AG0	0xD1		Stall on No AD Agent0 Transgress Credits
STALL1_NO_TxR_HORZ_CRD_AD_AG1	0xD3		Stall on No AD Agent1 Transgress Credits
STALL1_NO_TxR_HORZ_CRD_BL_AG0	0xD5		Stall on No BL Agent0 Transgress Credits
STALL1_NO_TxR_HORZ_CRD_BL_AG1	0xD7		Stall on No BL Agent1 Transgress Credits
TxR_HORZ_ADS_USED	0xA6		CMS Horizontal ADS Used
TxR_HORZ_BYPASS	0xA7		CMS Horizontal Bypass Used
TxR_HORZ_CYCLES_FULL	0xA2		Cycles CMS Horizontal Egress Queue is Full
TxR_HORZ_CYCLES_NE	0xA3		Cycles CMS Horizontal Egress Queue is Not Empty
TxR_HORZ_INSERTS	0xA1		CMS Horizontal Egress Inserts
TxR_HORZ_NACK	0xA4		CMS Horizontal Egress NACKs
TxR_HORZ_OCCUPANCY	0xA0		CMS Horizontal Egress Occupancy
TxR_HORZ_STARVED	0xA5		CMS Horizontal Egress Injection Starvation
TxR_VERT_ADS_USED	0x9C		CMS Vertical ADS Used
TxR_VERT_BYPASS	0x9D		CMS Vertical ADS Used
TxR_VERT_BYPASS_1	0x9E		CMS Vertical ADS Used
TxR_VERT_CYCLES_FULL0	0x94		Cycles CMS Vertical Egress Queue Is Full
TxR_VERT_CYCLES_FULL1	0x95		Cycles CMS Vertical Egress Queue Is Full
TxR_VERT_CYCLES_NE0	0x96		Cycles CMS Vertical Egress Queue Is Not Empty
TxR_VERT_CYCLES_NE1	0x97		Cycles CMS Vertical Egress Queue Is Not Empty
TxR_VERT_INSERTS0	0x92		CMS Vert Egress Allocations
TxR_VERT_INSERTS1	0x93		CMS Vert Egress Allocations
TxR_VERT_NACK0	0x98		CMS Vertical Egress NACKs
TxR_VERT_NACK1	0x99		CMS Vertical Egress NACKs
TxR_VERT_OCCUPANCY0	0x90		CMS Vert Egress Occupancy
TxR_VERT_OCCUPANCY1	0x91		CMS Vert Egress Occupancy
TxR_VERT_STARVED0	0x9A		CMS Vertical Egress Injection Starvation
TxR_VERT_STARVED1	0x9B		CMS Vertical Egress Injection Starvation
VERT_RING_AD_IN_USE	0xB0		Vertical AD Ring In Use
VERT_RING_AKC_IN_USE	0xB4		Vertical AKC Ring In Use
VERT_RING_AK_IN_USE	0xB1		Vertical AK Ring In Use

Table 2-1. Measured CMS Box Events (Sheet 3 of 3)

Symbol Name	Event Code	Ctrs	Description
VERT_RING_BL_IN_USE	0xB2		Vertical BL Ring in Use
VERT_RING_IV_IN_USE	0xB3		Vertical IV Ring in Use
VERT_RING_TGC_IN_USE	0xB5		Vertical TGC Ring In Use

2.1.3 CMS Box Performance Monitor Event List

The section enumerates 5th Gen Intel® Xeon® Scalable Processor performance monitoring events for the CMS box.

AG0_AD_CRD_ACQUIRED0

- **Title:**
- **Category:** Transgress Credit Events
- **Event Code:** . 0x80
- **Register Restrictions :**
- **Definition:** Number of CMS Agent 0 AD credits acquired in a given cycle, per transgress.

Table 2-2. Unit Masks for AG0_AD_CRD_ACQUIRED0

Extension	umask [15:8]	Description
TGR0	bxxxxxxx1	For Transgress 0
TGR1	bxxxxxx1x	For Transgress 1
TGR2	bxxxxx1xx	For Transgress 2
TGR3	bxxxx1xxx	For Transgress 3
TGR4	bxxx1xxxx	For Transgress 4
TGR5	bxx1xxxxx	For Transgress 5
TGR6	bx1xxxxxx	For Transgress 6
TGR7	b1xxxxxxx	For Transgress 7

AG0_AD_CRD_ACQUIRED1

- **Title:**
- **Category:** Transgress Credit Events
- **Event Code:** 0x81
- **Register Restrictions :**
- **Definition:** Number of CMS Agent 0 AD credits acquired in a given cycle, per transgress.

Table 2-3. Unit Masks for AG0_AD_CRD_ACQUIRED1

Extension	umask [15:8]	Description
TGR8	bxxxxxxx1	For Transgress 8
TGR9	bxxxxxx1x	For Transgress 9
TGR10	bxxxxx1xx	For Transgress 10
TGR11	bxxxx1xxx	For Transgress 11
TGR12	bxxx1xxxx	For Transgress 12
TGR13	bxx1xxxxx	For Transgress 13
TGR14	bx1xxxxxx	For Transgress 14
TGR15	b1xxxxxxx	For Transgress 15

AG0_AD_CRD_OCCUPANCY0

- **Title:**
- **Category:** Transgress Credit Events
- **Event Code:** 0x82
- **Register Restrictions :**
- **Definition:** Number of CMS Agent 0 AD credits in use in a given cycle, per transgress.

Table 2-4. Unit Masks for AG0_AD_CRD_OCCUPANCY0

Extension	umask [15:8]	Description
TGR0	b00000001	For Transgress 0
TGR1	b00000010	For Transgress 1
TGR2	b00000100	For Transgress 2
TGR3	b00001000	For Transgress 3
TGR4	b00010000	For Transgress 4
TGR5	b00100000	For Transgress 5
TGR6	b01000000	For Transgress 6
TGR7	b10000000	For Transgress 7

AG0_AD_CRD_OCCUPANCY1

- **Title:**
- **Category:** Transgress Credit Events
- **Event Code:** 0x83
- **Register Restrictions :**
- **Definition:** Number of CMS Agent 0 AD credits in use in a given cycle, per transgress.

Table 2-5. Unit Masks for AG0_AD_CRD_OCCUPANCY1

Extension	umask [15:8]	Description
TGR8	b00000001	For Transgress 8
TGR9	b00000010	For Transgress 9
TGR10	b00000100	For Transgress 10
TGR11	b00001000	For Transgress 11
TGR12	b00010000	For Transgress 12
TGR13	b00100000	For Transgress 13
TGR14	b01000000	For Transgress 14
TGR15	b10000000	For Transgress 15

AG0_BL_CRD_ACQUIRED0

- **Title:**
- **Category:** Transgress Credit Events
- **Event Code:** 0x88
- **Register Restrictions :**
- **Definition:** Number of CMS Agent 0 BL credits acquired in a given cycle, per transgress.

Table 2-6. Unit Masks for AG0_BL_CRD_ACQUIRED0 (Sheet 1 of 2)

Extension	umask [15:8]	Description
TGR0	bxxxxxxx1	For Transgress 0
TGR1	bxxxxxx1x	For Transgress 1
TGR2	bxxxxx1xx	For Transgress 2
TGR3	bxxxx1xxx	For Transgress 3
TGR4	bxxx1xxxx	For Transgress 4
TGR5	bxx1xxxxx	For Transgress 5

Table 2-6. Unit Masks for AG0_BL_CRD_ACQUIRED0 (Sheet 2 of 2)

Extension	umask [15:8]	Description
TGR6	bx1xxxxxx	For Transgress 6
TGR7	b1xxxxxxx	For Transgress 7

AG0_BL_CRD_ACQUIRED1

- **Title:**
- **Category:** Transgress Credit Events
- **Event Code:** 0x89
- **Register Restrictions :**
- **Definition:** Number of CMS Agent 0 BL credits acquired in a given cycle, per transgress.

Table 2-7. Unit Masks for AG0_BL_CRD_ACQUIRED1

Extension	umask [15:8]	Description
TGR8	bxxxxxxx1	For Transgress 8
TGR9	bxxxxxx1x	For Transgress 9
TGR10	bxxxxx1xx	For Transgress 10
TGR11	bxxxx1xxx	For Transgress 11
TGR12	bxxx1xxxx	For Transgress 12
TGR13	bxx1xxxxx	For Transgress 13
TGR14	bx1xxxxxx	For Transgress 14
TGR15	b1xxxxxxx	For Transgress 15

AG0_BL_CRD_OCCUPANCY0

- **Title:**
- **Category:** Transgress Credit Events
- **Event Code:** 0x8A
- **Register Restrictions :**
- **Definition:** Number of CMS Agent 0 BL credits in use in a given cycle, per transgress.

Table 2-8. Unit Masks for AG0_BL_CRD_OCCUPANCY0

Extension	umask [15:8]	Description
TGR0	b00000001	For Transgress 0
TGR1	b00000010	For Transgress 1
TGR2	b00000100	For Transgress 2
TGR3	b00001000	For Transgress 3
TGR4	b00010000	For Transgress 4
TGR5	b00100000	For Transgress 5
TGR6	b01000000	For Transgress 6
TGR7	b10000000	For Transgress 7

AG0_BL_CRD_OCCUPANCY1

- **Title:**
- **Category:** Transgress Credit Events
- **Event Code:** 0x8B
- **Register Restrictions :**
- **Definition:** Number of CMS Agent 0 BL credits in use in a given cycle, per transgress.

Table 2-9. Unit Masks for AG0_BL_CRD_OCCUPANCY1

Extension	umask [15:8]	Description
TGR8	b00000001	For Transgress 8
TGR9	b00000010	For Transgress 9
TGR10	b00000100	For Transgress 10
TGR11	b00001000	For Transgress 11
TGR12	b00010000	For Transgress 12
TGR13	b00100000	For Transgress 13
TGR14	b01000000	For Transgress 14
TGR15	b10000000	For Transgress 15

AG1_AD_CRD_ACQUIRED0

- **Title:**
- **Category:** Transgress Credit Events
- **Event Code:** 0x84
- **Register Restrictions :**
- **Definition:** Number of CMS Agent 1 AD credits acquired in a given cycle, per transgress.

Table 2-10. Unit Masks for AG1_AD_CRD_ACQUIRED0

Extension	umask [15:8]	Description
TGR0	bxxxxxxx1	For Transgress 0
TGR1	bxxxxxx1x	For Transgress 1
TGR2	bxxxxx1xx	For Transgress 2
TGR3	bxxxx1xxx	For Transgress 3
TGR4	bxxx1xxxx	For Transgress 4
TGR5	bxx1xxxxx	For Transgress 5
TGR6	bx1xxxxxx	For Transgress 6
TGR7	b1xxxxxxx	For Transgress 7

AG1_AD_CRD_ACQUIRED1

- **Title:**
- **Category:** Transgress Credit Events
- **Event Code:** 0x85
- **Register Restrictions :**
- **Definition:** Number of CMS Agent 1 AD credits acquired in a given cycle, per transgress.

Table 2-11. Unit Masks for AG1_AD_CRD_ACQUIRED1 (Sheet 1 of 2)

Extension	umask [15:8]	Description
TGR8	bxxxxxxx1	For Transgress 8
TGR9	bxxxxxx1x	For Transgress 9
TGR10	bxxxxx1xx	For Transgress 10
TGR11	bxxxx1xxx	For Transgress 11
TGR12	bxxx1xxxx	For Transgress 12
TGR13	bxx1xxxxx	For Transgress 13

Table 2-11. Unit Masks for AG1_AD_CRD_ACQUIRED1 (Sheet 2 of 2)

Extension	umask [15:8]	Description
TGR14	bx1xxxxxx	For Transgress 14
TGR15	b1xxxxxxx	For Transgress 15

AG1_AD_CRD_OCCUPANCY0

- **Title:**
- **Category:** Transgress Credit Events
- **Event Code:** 0x86
- **Register Restrictions :**
- **Definition:** Number of CMS Agent 1 AD credits in use in a given cycle, per transgress.

Table 2-12. Unit Masks for AG1_AD_CRD_OCCUPANCY0

Extension	umask [15:8]	Description
TGR0	b00000001	For Transgress 0
TGR1	b00000010	For Transgress 1
TGR2	b00000100	For Transgress 2
TGR3	b00001000	For Transgress 3
TGR4	b00010000	For Transgress 4
TGR5	b00100000	For Transgress 5
TGR6	b01000000	For Transgress 6
TGR7	b10000000	For Transgress 7

AG1_AD_CRD_OCCUPANCY1

- **Title:**
- **Category:** Transgress Credit Events
- **Event Code:** 0x87
- **Register Restrictions :**
- **Definition:** Number of CMS Agent 1 AD credits in use in a given cycle, per transgress.

Table 2-13. Unit Masks for AG1_AD_CRD_OCCUPANCY1

Extension	umask [15:8]	Description
TGR8	b00000001	For Transgress 8
TGR9	b00000010	For Transgress 9
TGR10	b00000100	For Transgress 10
TGR11	b00001000	For Transgress 11
TGR12	b00010000	For Transgress 12
TGR13	b00100000	For Transgress 13
TGR14	b01000000	For Transgress 14
TGR15	b10000000	For Transgress 15

AG1_BL_CRD_ACQUIRED0

- **Title:**
- **Category:** Transgress Credit Events
- **Event Code:** 0x8C
- **Register Restrictions :**
- **Definition:** Number of CMS Agent 1 BL credits acquired in a given cycle, per transgress.

Table 2-14. Unit Masks for AG1_BL_CRD_ACQUIRED0

Extension	umask [15:8]	Description
TGR0	bxxxxxxx1	For Transgress 0
TGR1	bxxxxxx1x	For Transgress 1
TGR2	bxxxxx1xx	For Transgress 2
TGR3	bxxxx1xxx	For Transgress 3
TGR4	bxxx1xxxx	For Transgress 4
TGR5	bxx1xxxxx	For Transgress 5
TGR6	bx1xxxxxx	For Transgress 4
TGR7	b1xxxxxxx	For Transgress 5

AG1_BL_CRD_ACQUIRED1

- **Title:**
- **Category:** Transgress Credit Events
- **Event Code:** 0x8D
- **Register Restrictions :**
- **Definition:** Number of CMS Agent 1 BL credits acquired in a given cycle, per transgress.

Table 2-15. Unit Masks for AG1_BL_CRD_ACQUIRED1

Extension	umask [15:8]	Description
TGR8	bxxxxxxx1	For Transgress 8
TGR9	bxxxxxx1x	For Transgress 9
TGR10	bxxxxx1xx	For Transgress 10
TGR11	bxxxx1xxx	For Transgress 11
TGR12	bxxx1xxxx	For Transgress 12
TGR13	bxx1xxxxx	For Transgress 13
TGR14	bx1xxxxxx	For Transgress 14
TGR15	b1xxxxxxx	For Transgress 15

AG1_BL_CRD_OCCUPANCY0

- **Title:**
- **Category:** Transgress Credit Events
- **Event Code:** 0x8E
- **Register Restrictions :**
- **Definition:** Number of CMS Agent 1 BL credits in use in a given cycle, per transgress.

Table 2-16. Unit Masks for AG1_BL_CRD_OCCUPANCY0 (Sheet 1 of 2)

Extension	umask [15:8]	Description
TGR0	b00000001	For Transgress 0
TGR1	b00000010	For Transgress 1
TGR2	b00000100	For Transgress 2
TGR3	b00001000	For Transgress 3
TGR4	b00010000	For Transgress 4
TGR5	b00100000	For Transgress 5

Table 2-16. Unit Masks for AG1_BL_CRD_OCCUPANCY0 (Sheet 2 of 2)

Extension	umask [15:8]	Description
TGR6	b01000000	For Transgress 6
TGR7	b10000000	For Transgress 7

AG1_BL_CRD_OCCUPANCY1

- **Title:**
- **Category:** Transgress Credit Events
- **Event Code:** 0x8F
- **Register Restrictions :**
- **Definition:** Number of CMS Agent 1 BL credits in use in a given cycle, per transgress.

Table 2-17. Unit Masks for AG1_BL_CRD_OCCUPANCY1

Extension	umask [15:8]	Description
TGR8	b00000001	For Transgress 8
TGR9	b00000010	For Transgress 9
TGR10	b00000100	For Transgress 10
TGR11	b00001000	For Transgress 11
TGR12	b00010000	For Transgress 12
TGR13	b00100000	For Transgress 13
TGR14	b01000000	For Transgress 14
TGR15	b10000000	For Transgress 15

CMS_CLOCKTICKS

- **Title:**
- **Category:** Misc Events
- **Event Code:** 0xC0
- **Register Restrictions :**
- **Definition:**

DISTRESS_ASSERTED

- **Title:**
- **Category:** Horizontal Ring Events
- **Event Code:** 0xAF
- **Register Restrictions :**

- **Definition:** Counts the number of cycles either the local or incoming distress signals are asserted.

Table 2-18. Unit Masks for DISTRESS_ASSERTED

Extension	umask [15:8]	Description
VERT	b00000001	Vertical If IRQ egress is full, then agents will throttle outgoing AD IDI transactions
HORZ	b00000010	Horizontal If TGR egress is full, then agents will throttle outgoing AD IDI transactions
DPT_LOCAL	bxxxxx1xx	DPT Local Dynamic Prefetch Throttle triggered by this tile
DPT_NONLOCAL	bxxxx1xxx	DPT Remote Dynamic Prefetch Throttle received by this tile
PMM_LOCAL	bxxx1xxxx	DDRT Local If the CHA TOR has too many PMM transactions, this signal will throttle outgoing MS2IDI traffic
PMM_NONLOCAL	bxx1xxxxx	DDRT Remote If another CHA TOR has too many PMM transactions, this signal will throttle outgoing MS2IDI traffic
DPT_STALL_IV	bx1xxxxxx	DPT Stalled - IV DPT occurred while regular IVs were received, causing DPT to be stalled
DPT_STALL_NOCRD	b1xxxxxxx	DPT Stalled - No Credit DPT occurred while credit not available causing DPT to be stalled

EGRESS_ORDERING

- **Title:**
- **Category:** Horizontal In Use Ring Events
- **Event Code:** 0xBA
- **Register Restrictions :**
- **Definition:** Counts number of cycles IV was blocked in the TGR egress due to SNP/GO Ordering requirements.

Table 2-19. Unit Masks for EGRESS_ORDERING

Extension	umask [15:8]	Description
IV_SNOOPGO_UP	bxxxxxxx1	Up
IV_SNOOPGO_DN	bxxxxx1xx	Down

HORZ_RING_AD_IN_USE

- **Title:**
- **Category:** Horizontal In Use Ring Events
- **Event Code:** 0xB6
- **Register Restrictions :**
- **Definition:** Counts the number of cycles that the horizontal AD ring is being used at this ring stop. This includes when packets are passing by and when packets are being

sunk, but does not include when packets are being sent from the ring stop. We really have two rings -- a clockwise ring and a counter-clockwise ring. On the left side of the ring, the "UP" direction is on the clockwise ring and "DN" is on the counter-clockwise ring. On the right side of the ring, this is reversed. The first half of the CBos are on the left side of the ring, and the second half are on the right side of the ring. In other words (for example), in a 4c part, Cbo 0 UP AD is NOT the same ring as CBo 2 UP AD because they are on opposite sides of the ring.

Table 2-20. Unit Masks for HORZ_RING_AD_IN_USE

Extension	umask [15:8]	Description
LEFT_EVEN	bxxxxxxx1	Left and Even
LEFT_ODD	bxxxxxx1x	Left and Odd
RIGHT_EVEN	bxxxxx1xx	Right and Even
RIGHT_ODD	bxxxx1xxx	Right and Odd

HORZ_RING_AKC_IN_USE

- **Title:**
- **Category:** Horizontal In Use Ring Events
- **Event Code:** 0xBB
- **Register Restrictions :**
- **Definition:** Counts the number of cycles that the Horizontal AKC ring is being used at this ring stop. This includes when packets are passing by and when packets are being sunk, but does not include when packets are being sent from the ring stop.

Table 2-21. Unit Masks for HORZ_RING_AKC_IN_USE

Extension	umask [15:8]	Description
LEFT_EVEN	bxxxxxxx1	Left and Even
LEFT_ODD	bxxxxxx1x	Left and Odd
RIGHT_EVEN	bxxxxx1xx	Right and Even
RIGHT_ODD	bxxxx1xxx	Right and Odd

HORZ_RING_AK_IN_USE

- **Title:**
- **Category:** Horizontal In Use Ring Events
- **Event Code:** 0xB7
- **Register Restrictions :**
- **Definition:** Counts the number of cycles that the horizontal AK ring is being used at this ring stop. This includes when packets are passing by and when packets are being sunk, but does not include when packets are being sent from the ring stop. We really have two rings -- a clockwise ring and a counter-clockwise ring. On the left side of the ring, the "UP" direction is on the clockwise ring and "DN" is on the counter-clockwise ring. On the right side of the ring, this is reversed. The first half of the CBos are on the left side of the ring, and the second half are on the right side of the ring.

In other words (for example), in a 4c part, Cbo 0 UP AD is NOT the same ring as CBo 2 UP AD because they are on opposite sides of the ring.

Table 2-22. Unit Masks for HORZ_RING_AK_IN_USE

Extension	umask [15:8]	Description
LEFT_EVEN	bxxxxxxx1	Left and Even
LEFT_ODD	bxxxxxx1x	Left and Odd
RIGHT_EVEN	bxxxxx1xx	Right and Even
RIGHT_ODD	bxxxx1xxx	Right and Odd

HORZ_RING_BL_IN_USE

- **Title:**
- **Category:** Horizontal In Use Ring Events
- **Event Code:** 0xB8
- **Register Restrictions :**
- **Definition:** Counts the number of cycles that the horizontal BL ring is being used at this ring stop. This includes when packets are passing by and when packets are being sunk, but does not include when packets are being sent from the ring stop. We really have two rings -- a clockwise ring and a counter-clockwise ring. On the left side of the ring, the "UP" direction is on the clockwise ring and "DN" is on the counter-clockwise ring. On the right side of the ring, this is reversed. The first half of the CBos are on the left side of the ring, and the second half are on the right side of the ring. In other words (for example), in a 4c part, Cbo 0 UP AD is NOT the same ring as CBo 2 UP AD because they are on opposite sides of the ring.

Table 2-23. Unit Masks for HORZ_RING_BL_IN_USE

Extension	umask [15:8]	Description
LEFT_EVEN	bxxxxxxx1	Left and Even
LEFT_ODD	bxxxxxx1x	Left and Odd
RIGHT_EVEN	bxxxxx1xx	Right and Even
RIGHT_ODD	bxxxx1xxx	Right and Odd

HORZ_RING_IV_IN_USE

- **Title:**
- **Category:** Horizontal In Use Ring Events
- **Event Code:** 0xB9
- **Register Restrictions :**
- **Definition:** Counts the number of cycles that the horizontal IV ring is being used at this ring stop. This includes when packets are passing by and when packets are being sunk, but does not include when packets are being sent from the ring stop. There is only 1 IV ring. Therefore, if one wants to monitor the "Even" ring, they should select both UP_EVEN and DN_EVEN. To monitor the "Odd" ring, they should select both UP_ODD and DN_ODD.

Table 2-24. Unit Masks for HORZ_RING_IV_IN_USE

Extension	umask [15:8]	Description
LEFT	bxxxxxxx1	Left
RIGHT	bxxxxx1xx	Right

MISC_EXTERNAL

- **Title:**
- **Category:** External Misc Events (for example, From MS2IDI)
- **Event Code:** 0xE6
- **Register Restrictions :**
- **Definition:**

Table 2-25. Unit Masks for MISC_EXTERNAL

Extension	umask [15:8]	Description
MBE_INST0	bxxxxxxx1	Number of cycles MBE is high for MS2IDI0
MBE_INST1	bxxxxxx1x	Number of cycles MBE is high for MS2IDI1

RING_BOUNCES_HORZ

- **Title:**
- **Category:** Horizontal Ring Events
- **Event Code:** 0xAC
- **Register Restrictions :**
- **Definition:** Number of cycles incoming messages from the horizontal ring that were bounced, by ring type.

Table 2-26. Unit Masks for RING_BOUNCES_HORZ

Extension	umask [15:8]	Description
AD	bxxxxxxx1	AD
AK	bxxxxxx1x	AK
BL	bxxxxx1xx	BL
IV	bxxxx1xxx	IV

RING_BOUNCES_VERT

- **Title:**
- **Category:** Vertical Ring Events
- **Event Code:** 0xAA
- **Register Restrictions :**
- **Definition:** Number of cycles incoming messages from the vertical ring that were bounced, by ring type.

Table 2-27. Unit Masks for RING_BOUNCES_VERT

Extension	umask [15:8]	Description
AD	bxxxxxxx1	AD
AK	bxxxxxx1x	Acknowledgments to core
BL	bxxxxx1xx	Data Responses to core
IV	bxxxx1xxx	Snoops of processor’s cache.'
AKC	bxxx1xxxx	

RING_SINK_STARVED_HORZ

- **Title:**
- **Category:** Horizontal Ring Events
- **Event Code:** 0xAD
- **Register Restrictions :**
- **Definition:**

Table 2-28. Unit Masks for RING_SINK_STARVED_HORZ

Extension	umask [15:8]	Description
AD	bxxxxxxx1	AD
AK	bxxxxxx1x	AK
BL	bxxxxx1xx	BL
IV	bxxxx1xxx	IV
AK_AG1	bxx1xxxxx	Acknowledgments to Agent 1

RING_SINK_STARVED_VERT

- **Title:**
- **Category:** Vertical Ring Events
- **Event Code:** 0xAB
- **Register Restrictions :**
- **Definition:**

Table 2-29. Unit Masks for RING_SINK_STARVED_VERT

Extension	umask [15:8]	Description
AD	bxxxxxxx1	AD
AK	bxxxxxx1x	Acknowledgments to core
BL	bxxxxx1xx	Data Responses to core
IV	bxxxx1xxx	Snoops of processor's cache.'
AKC	bxxx1xxxx	

RING_SRC_THRTL

- **Title:**
- **Category:** Horizontal Ring Events
- **Event Code:** 0xAE
- **Register Restrictions :**
- **Definition:**

RxR_BUSY_STARVED

- **Title:**
- **Category:** Ingress Transgress Events
- **Event Code:** 0xE5
- **Register Restrictions :**
- **Definition:** Counts cycles under injection starvation mode. This starvation is triggered when the CMS ingress cannot send a transaction onto the mesh for a long period of time. In this case, because a message from the other queue has higher priority.

Table 2-30. Unit Masks for RxR_BUSY_STARVED

Extension	umask [15:8]	Description
AD_UNCRD	b00000001	AD - Uncredited
BL_UNCRD	b00000100	BL - Uncredited
AD_CRD	b00010000	AD - Credited
AD_ALL	b00010001	AD - All All == Credited + Uncredited
BL_CRD	b01000000	BL - Credited
BL_ALL	b01000100	BL - All All == Credited + Uncredited

RxR_BYPASS

- **Title:**
- **Category:** Ingress Transgress Events
- **Event Code:** 0xE2
- **Register Restrictions :**
- **Definition:** Number of packets bypassing the CMS Ingress

Table 2-31. Unit Masks for RxR_BYPASS

Extension	umask [15:8]	Description
AD_UNCRD	b00000001	AD - Uncredited
AK	b00000010	AK
BL_UNCRD	b00000100	BL - Uncredited
IV	b00001000	IV
AD_CRD	b00010000	AD - Credited
AD_ALL	b00010001	AD - All All == Credited + Uncredited
BL_CRD	b01000000	BL - Credited
BL_ALL	b01000100	BL - All All == Credited + Uncredited
AKC_UNCRD	b10000000	AKC - Uncredited

RxR_CRD_STARVED

- **Title:**
- **Category:** Ingress Transgress Events
- **Event Code:** 0xE3
- **Register Restrictions :**
- **Definition:** Counts cycles under injection starvation mode. This starvation is triggered when the CMS ingress cannot send a transaction onto the mesh for a long period of time. In this case, the Ingress is unable to forward to the egress due to a lack of credit.

Table 2-32. Unit Masks for RxR_CRD_STARVED (Sheet 1 of 2)

Extension	umask [15:8]	Description
AD_UNCRD	b00000001	AD - Uncredited
AK	b00000010	AK
BL_UNCRD	b00000100	BL - Uncredited
IV	b00001000	IV

Table 2-32. Unit Masks for RxR_CRD_STARVED (Sheet 2 of 2)

Extension	umask [15:8]	Description
AD_CRD	b00010000	AD - Credited
AD_ALL	b00010001	AD - All All == Credited + Uncredited
BL_CRD	b01000000	BL - Credited
BL_ALL	b01000100	BL - All All == Credited + Uncredited
IFV	b10000000	IFV - Credited

RxR_CRD_STARVED_1

- **Title:**
- **Category:** Ingress Transgress Events
- **Event Code:** 0xE4
- **Register Restrictions :**
- **Definition:** Counts cycles under injection starvation mode. This starvation is triggered when the CMS Ingress cannot send a transaction onto the mesh for a long period of time. In this case, the Ingress is unable to forward to the egress due to a lack of credit.

RxR_INSERTS

- **Title:**
- **Category:** Ingress Transgress Events
- **Event Code:** 0xE1
- **Register Restrictions :**
- **Definition:** Number of allocations into the CMS ingress. The ingress is used to queue up requests received from the mesh.

Table 2-33. Unit Masks for RxR_INSERTS (Sheet 1 of 2)

Extension	umask [15:8]	Description
AD_UNCRD	b00000001	AD - Uncredited
AK	b00000010	AK
BL_UNCRD	b00000100	BL - Uncredited
IV	b00001000	IV
AD_CRD	b00010000	AD - Credited
AD_ALL	b00010001	AD - All All == Credited + Uncredited
BL_CRD	b01000000	BL - Credited

Table 2-33. Unit Masks for RxR_INSERTS (Sheet 2 of 2)

Extension	umask [15:8]	Description
BL_ALL	b01000100	BL - All All == Credited + Uncredited
AKC_UNCRD	b10000000	AKC - Uncredited

RxR_OCCUPANCY

- **Title:**
- **Category:** Ingress Transgress Events
- **Event Code:** 0xE0
- **Register Restrictions :**
- **Definition:** Occupancy event for the ingress buffers in the CMS. The ingress is used to queue up requests received from the mesh.

Table 2-34. Unit Masks for RxR_OCCUPANCY

Extension	umask [15:8]	Description
AD_UNCRD	b00000001	AD - Uncredited
AK	b00000010	AK
BL_UNCRD	b00000100	BL - Uncredited
IV	b00001000	IV
AD_CRD	b00010000	AD - Credited
AD_ALL	b00010001	AD - All All == Credited + Uncredited
BL_CRD	b00100000	BL - Credited
BL_ALL	b01000100	BL - All All == Credited + Uncredited
AKC_UNCRD	b10000000	AKC - Uncredited

STALL0_NO_TxR_HORZ_CRD_AD_AG0

- **Title:**
- **Category:** Transgress Credit Events
- **Event Code:** 0xD0
- **Register Restrictions :**
- **Definition:** Number of cycles the AD agent 0 egress buffer is stalled waiting for a TGR credit to become available, per transgress.

Table 2-35. Unit Masks for STALL0_NO_TxR_HORZ_CRD_AD_AG0

Extension	umask [15:8]	Description
TGR0	bxxxxxxx1	For Transgress 0
TGR1	bxxxxxx1x	For Transgress 1
TGR2	bxxxxx1xx	For Transgress 2
TGR3	bxxxx1xxx	For Transgress 3
TGR4	bxxx1xxxx	For Transgress 4
TGR5	bxx1xxxxx	For Transgress 5
TGR6	bx1xxxxxx	For Transgress 6
TGR7	b1xxxxxxx	For Transgress 7

STALL0_NO_TxR_HORZ_CRD_AD_AG1

- **Title:**
- **Category:** Transgress Credit Events
- **Event Code:** 0xD2
- **Register Restrictions :**
- **Definition:** Number of cycles the AD agent 1 egress buffer is stalled waiting for a TGR credit to become available, per transgress.

Table 2-36. Unit Masks for STALL0_NO_TxR_HORZ_CRD_AD_AG1

Extension	umask [15:8]	Description
TGR0	bxxxxxxx1	For Transgress 0
TGR1	bxxxxxx1x	For Transgress 1
TGR2	bxxxxx1xx	For Transgress 2
TGR3	bxxxx1xxx	For Transgress 3
TGR4	bxxx1xxxx	For Transgress 4
TGR5	bxx1xxxxx	For Transgress 5
TGR6	bx1xxxxxx	For Transgress 6
TGR7	b1xxxxxxx	For Transgress 7

STALL0_NO_TxR_HORZ_CRD_BL_AG0

- **Title:**
- **Category:** Transgress Credit Events
- **Event Code:** 0xD4
- **Register Restrictions :**
- **Definition:** Number of cycles the BL agent 0 egress buffer is stalled waiting for a TGR credit to become available, per transgress.

Table 2-37. Unit Masks for STALL0_NO_TxR_HORZ_CRD_BL_AG0

Extension	umask [15:8]	Description
TGR0	bxxxxxxx1	For Transgress 0
TGR1	bxxxxxx1x	For Transgress 1
TGR2	bxxxxx1xx	For Transgress 2
TGR3	bxxxx1xxx	For Transgress 3
TGR4	bxxx1xxxx	For Transgress 4
TGR5	bxx1xxxxx	For Transgress 5
TGR6	bx1xxxxxx	For Transgress 6
TGR7	b1xxxxxxx	For Transgress 7

STALL0_NO_TxR_HORZ_CRD_BL_AG1

- **Title:**
- **Category:** Transgress Credit Events
- **Event Code:** 0xD6
- **Register Restrictions :**
- **Definition:** Number of cycles the BL agent 1 egress buffer is stalled waiting for a TGR credit to become available, per transgress.

Table 2-38. Unit Masks for STALL0_NO_TxR_HORZ_CRD_BL_AG1 (Sheet 1 of 2)

Extension	umask [15:8]	Description
TGR0	bxxxxxxx1	For Transgress 0
TGR1	bxxxxxx1x	For Transgress 1
TGR2	bxxxxx1xx	For Transgress 2
TGR3	bxxxx1xxx	For Transgress 3
TGR4	bxxx1xxxx	For Transgress 4
TGR5	bxx1xxxxx	For Transgress 5

Table 2-38. Unit Masks for STALL0_NO_TxR_HORZ_CRD_BL_AG1 (Sheet 2 of 2)

Extension	umask [15:8]	Description
TGR6	bx1xxxxxx	For Transgress 6
TGR7	b1xxxxxxx	For Transgress 7

STALL1_NO_TxR_HORZ_CRD_AD_AG0

- **Title:**
- **Category:** Transgress Credit Events
- **Event Code:** 0xD1
- **Register Restrictions :**
- **Definition:** Number of cycles the AD agent 0 egress buffer is stalled waiting for a TGR credit to become available, per transgress.

Table 2-39. Unit Masks for STALL1_NO_TxR_HORZ_CRD_AD_AG0

Extension	umask [15:8]	Description
TGR8	bxxxxxxx1	For Transgress 8
TGR9	bxxxxxx1x	For Transgress 9
TGR10	bxxxxx1xx	For Transgress 10
TGR11	bxxxx1xxx	For Transgress 11
TGR12	bxxx1xxxx	For Transgress 12
TGR13	bxx1xxxxx	For Transgress 13
TGR14	bx1xxxxxx	For Transgress 14
TGR15	b1xxxxxxx	For Transgress 15

STALL1_NO_TxR_HORZ_CRD_AD_AG1

- **Title:**
- **Category:** Transgress Credit Events
- **Event Code:** 0xD3
- **Register Restrictions :**
- **Definition:** Number of cycles the AD agent 1 egress buffer is stalled waiting for a TGR credit to become available, per transgress.

Table 2-40. Unit Masks for STALL1_NO_TxR_HORZ_CRD_AD_AG1

Extension	umask [15:8]	Description
TGR8	bxxxxxxx1	For Transgress 8
TGR9	bxxxxxx1x	For Transgress 9
TGR10	bxxxxx1xx	For Transgress 10
TGR11	bxxxx1xxx	For Transgress 11
TGR12	bxxx1xxxx	For Transgress 12
TGR13	bxx1xxxxx	For Transgress 13
TGR14	bx1xxxxxx	For Transgress 14
TGR15	b1xxxxxxx	For Transgress 15

STALL1_NO_TxR_HORZ_CRD_BL_AG0

- **Title:**
- **Category:** Transgress Credit Events
- **Event Code:** 0xD5
- **Register Restrictions :**
- **Definition:** Number of cycles the BL agent 0 egress buffer is stalled waiting for a TGR credit to become available, per transgress.

Table 2-41. Unit Masks for STALL1_NO_TxR_HORZ_CRD_BL_AG0

Extension	umask [15:8]	Description
TGR8	bxxxxxxx1	For Transgress 8
TGR9	bxxxxxx1x	For Transgress 9
TGR10	bxxxxx1xx	For Transgress 10
TGR11	bxxxx1xxx	For Transgress 11
TGR12	bxxx1xxxx	For Transgress 12
TGR13	bxx1xxxxx	For Transgress 13
TGR14	bx1xxxxxx	For Transgress 14
TGR15	b1xxxxxxx	For Transgress 15

STALL1_NO_TxR_HORZ_CRD_BL_AG1

- **Title:**
- **Category:** Transgress Credit Events
- **Event Code:** 0xD7
- **Register Restrictions :**
- **Definition:** Number of cycles the BL agent 1 egress buffer is stalled waiting for a TGR credit to become available, per transgress.

Table 2-42. Unit Masks for STALL1_NO_TxR_HORZ_CRD_BL_AG1

Extension	umask [15:8]	Description
TGR8	bxxxxxxx1	For Transgress 8
TGR9	bxxxxxx1x	For Transgress 9
TGR10	bxxxxx1xx	For Transgress 10
TGR11	bxxxx1xxx	For Transgress 11
TGR12	bxxx1xxxx	For Transgress 12
TGR13	bxx1xxxxx	For Transgress 13
TGR14	bx1xxxxxx	For Transgress 14
TGR15	b1xxxxxxx	For Transgress 15

TxR_HORZ_ADS_USED

- **Title:**
- **Category:** Horizontal Egress Events
- **Event Code:** 0xA6
- **Register Restrictions :**
- **Definition:** Number of packets using the horizontal anti-deadlock slot, broken down by ring type and CMS agent.

Table 2-43. Unit Masks for TxR_HORZ_ADS_USED

Extension	umask [15:8]	Description
AD_UNCRD	b00000001	AD - Uncredited
BL_UNCRD	b00000100	BL - Uncredited
AD_CRD	b00010000	AD - Credited
AD_ALL	b00010001	AD - All All == Credited + Uncredited
BL_CRD	b01000000	BL - Credited
BL_ALL	b01000100	BL - All All == Credited + Uncredited

TxR_HORZ_BYPASS

- **Title:**
- **Category:** Horizontal Egress Events
- **Event Code:** 0xA7
- **Register Restrictions :**
- **Definition:** Number of packets bypassing the horizontal egress, broken down by ring type and CMS agent.

Table 2-44. Unit Masks for TxR_HORZ_BYPASS

Extension	umask [15:8]	Description
AD_UNCRD	b00000001	AD - Uncredited
AK	b00000010	AK
BL_UNCRD	b00000100	BL - Uncredited
IV	b00001000	IV
AD_CRD	b00010000	AD - Credited
AD_ALL	b00010001	AD - All All == Credited + Uncredited
BL_CRD	b01000000	BL - Credited
BL_ALL	b01000100	BL - All All == Credited + Uncredited
AKC_UNCRD	b10000000	AKC - Uncredited

TxR_HORZ_CYCLES_FULL

- **Title:**
- **Category:** Horizontal Egress Events
- **Event Code:** 0xA2
- **Register Restrictions :**
- **Definition:** Cycles the transgress buffers in the common mesh stop are full. The egress is used to queue up requests destined for the horizontal ring on the mesh.

Table 2-45. Unit Masks for TxR_HORZ_CYCLES_FULL (Sheet 1 of 2)

Extension	umask [15:8]	Description
AD_UNCRD	b00000001	AD - Uncredited
AK	b00000010	AK
BL_UNCRD	b00000100	BL - Uncredited
IV	b00001000	IV
AD_CRD	b00010000	AD - Credited

Table 2-45. Unit Masks for TxR_HORZ_CYCLES_FULL (Sheet 2 of 2)

Extension	umask [15:8]	Description
AD_ALL	b00010001	AD - All All == Credited + Uncredited
BL_CRD	b01000000	BL - Credited
BL_ALL	b01000100	BL - All All == Credited + Uncredited
AKC_UNCRD	b10000000	AKC - Uncredited

TxR_HORZ_CYCLES_NE

- **Title:**
- **Category:** Horizontal Egress Events
- **Event Code:** 0xA3
- **Register Restrictions :**
- **Definition:** Cycles the Transgress buffers in the common mesh stop are not-empty. The egress is used to queue up requests destined for the horizontal ring on the mesh.

Table 2-46. Unit Masks for TxR_HORZ_CYCLES_NE

Extension	umask [15:8]	Description
AD_UNCRD	b00000001	AD - Uncredited
AK	b00000010	AK
BL_UNCRD	b00000100	BL - Uncredited
IV	b00001000	IV
AD_CRD	b00010000	AD - Credited
AD_ALL	b00010001	AD - All All == Credited + Uncredited
BL_CRD	b01000000	BL - Credited
BL_ALL	b01000100	BL - All All == Credited + Uncredited
AKC_UNCRD	b10000000	AKC - Uncredited

TxR_HORZ_INSERTS

- **Title:**
- **Category:** Horizontal Egress Events
- **Event Code:** 0xA1
- **Register Restrictions :**
- **Definition:** Number of allocations into the Transgress buffers in the common mesh stop. The egress is used to queue up requests destined for the horizontal ring on the mesh.

Table 2-47. Unit Masks for TxR_HORZ_INSERTS

Extension	umask [15:8]	Description
AD_UNCRD	b00000001	AD - Uncredited
AK	b00000010	AK
BL_UNCRD	b00000100	BL - Uncredited
IV	b00001000	IV
AD_CRD	b00010000	AD - Credited
AD_ALL	b00010001	AD - All All == Credited + Uncredited
BL_CRD	b01000000	BL - Credited
BL_ALL	b01000100	BL - All All == Credited + Uncredited
AKC_UNCRD	b10000000	AKC - Uncredited

TxR_HORZ_NACK

- **Title:**
- **Category:** Horizontal Egress Events
- **Event Code:** 0xA4
- **Register Restrictions :**
- **Definition:** Counts number of egress packets NACKed onto the horizontal ring.

Table 2-48. Unit Masks for TxR_HORZ_NACK

Extension	umask [15:8]	Description
AD_UNCRD	b00000001	AD - Uncredited
AK	b00000010	AK
BL_UNCRD	b00000100	BL - Uncredited
IV	b00001000	IV
AD_CRD	b00010000	AD - Credited
AD_ALL	b00010001	AD - All All == Credited + Uncredited
BL_CRD	b01000000	BL - Credited
BL_ALL	b01000100	BL - All All == Credited + Uncredited
AKC_UNCRD	b10000000	AKC - Uncredited

intel.

TxR_HORZ_OCCUPANCY

- **Title:**
- **Category:** Horizontal Egress Events
- **Event Code:** 0xA0
- **Register Restrictions :**
- **Definition:** Occupancy event for the transgress buffers in the common mesh stop. The egress is used to queue up requests destined for the horizontal ring on the mesh.

Table 2-49. Unit Masks for TxR_HORZ_OCCUPANCY

Extension	umask [15:8]	Description
AD_UNCRD	b00000001	AD - Uncredited
AK	b00000010	AK
BL_UNCRD	b00000100	BL - Uncredited
IV	b00001000	IV
AD_CRD	b00010000	AD - Credited
AD_ALL	b00010001	AD - All All == Credited + Uncredited
BL_CRD	b01000000	BL - Credited
BL_ALL	b01000100	BL - All All == Credited + Uncredited
AKC_UNCRD	b10000000	AKC - Uncredited

TxR_HORZ_STARVED

- **Title:**
- **Category:** Horizontal Egress Events
- **Event Code:** 0xA5
- **Register Restrictions :**
- **Definition:** Counts injection starvation. This starvation is triggered when the CMS transgress buffer cannot send a transaction onto the horizontal ring for a long period of time.

Table 2-50. Unit Masks for TxR_HORZ_STARVED (Sheet 1 of 2)

Extension	umask [15:8]	Description
AD_ALL	b00000001	AD - All All == Credited + Uncredited
AD_UNCRD	b00000001	AD - Uncredited
AK	b00000010	AK
BL_ALL	b00000100	BL - All All == Credited + Uncredited

Table 2-50. Unit Masks for TxR_HORZ_STARVED (Sheet 2 of 2)

Extension	umask [15:8]	Description
BL_UNCRD	b00000100	BL - Uncredited
IV	b00001000	IV
AKC_UNCRD	b10000000	AKC - Uncredited

TxR_VERT_ADS_USED

- **Title:**
- **Category:** Vertical Egress Events
- **Event Code:** 0x9C
- **Register Restrictions :**
- **Definition:** Number of packets using the vertical anti-deadlock slot, broken down by ring type and CMS agent.

Table 2-51. Unit Masks for TxR_VERT_ADS_USED

Extension	umask [15:8]	Description
AD_AG0	bxxxxxxx1	AD - Agent 0
BL_AG0	bxxxxx1xx	BL - Agent 0
AD_AG1	bxxx1xxxx	AD - Agent 1
BL_AG1	bx1xxxxxx	BL - Agent 1

TxR_VERT_BYPASS

- **Title:**
- **Category:** Vertical Egress Events
- **Event Code:** 0x9D
- **Register Restrictions :**
- **Definition:** Number of packets bypassing the vertical egress, broken down by ring type and CMS agent.

Table 2-52. Unit Masks for TxR_VERT_BYPASS (Sheet 1 of 2)

Extension	umask [15:8]	Description
AD_AG0	bxxxxxxx1	AD - Agent 0
AK_AG0	bxxxxxx1x	AK - Agent 0
BL_AG0	bxxxxx1xx	BL - Agent 0
IV_AG1	bxxxx1xxx	IV - Agent 1

Table 2-52. Unit Masks for TxR_VERT_BYPASS (Sheet 2 of 2)

Extension	umask [15:8]	Description
AD_AG1	bxxx1xxxx	AD - Agent 1
AK_AG1	bxx1xxxxx	AK - Agent 1
BL_AG1	bx1xxxxxx	BL - Agent 1

TxR_VERT_BYPASS_1

- **Title:**
- **Category:** Vertical Egress Events
- **Event Code:** 0x9E
- **Register Restrictions :**
- **Definition:** Number of packets bypassing the vertical egress, broken down by ring type and CMS agent.

Table 2-53. Unit Masks for TxR_VERT_BYPASS_1

Extension	umask [15:8]	Description
AKC_AG0	bxxxxxxx1	AKC - Agent 0
AKC_AG1	bxxxxxx1x	AKC - Agent 1

TxR_VERT_CYCLES_FULL0

- **Title:**
- **Category:** Vertical Egress Events
- **Event Code:** 0x94
- **Register Restrictions :**
- **Definition:** Number of cycles the common mesh stop egress was not full. The egress is used to queue up requests destined for the vertical ring on the mesh.

Table 2-54. Unit Masks for TxR_VERT_CYCLES_FULL0 (Sheet 1 of 2)

Extension	umask [15:8]	Description
AD_AG0	bxxxxxxx1	AD - Agent 0 Ring transactions from Agent 0 destined for the AD ring. Some example include outbound requests, snoop requests, and snoop responses.
AK_AG0	bxxxxxx1x	AK - Agent 0 Ring transactions from Agent 0 destined for the AK ring. This is commonly used for credit returns and GO responses.
BL_AG0	bxxxxx1xx	BL - Agent 0 Ring transactions from Agent 0 destined for the BL ring. This is commonly used to send data from the cache to various destinations.
IV_AG0	bxxxx1xxx	IV - Agent 0 Ring transactions from Agent 0 destined for the IV ring. This is commonly used for snoops to the cores.

Table 2-54. Unit Masks for TxR_VERT_CYCLES_FULL0 (Sheet 2 of 2)

Extension	umask [15:8]	Description
AD_AG1	bxxx1xxxx	AD - Agent 1 Ring transactions from Agent 1 destined for the AD ring. This is commonly used for outbound requests.
AK_AG1	bxx1xxxxx	AK - Agent 1 Ring transactions from Agent 1 destined for the AK ring.
BL_AG1	bx1xxxxxx	BL - Agent 1 Ring transactions from Agent 1 destined for the BL ring. This is commonly used for transferring write back data to the cache.

TxR_VERT_CYCLES_FULL1

- **Title:**
- **Category:** Vertical Egress Events
- **Event Code:** 0x95
- **Register Restrictions :**
- **Definition:** Number of cycles the common mesh stop egress was not full. The egress is used to queue up requests destined for the vertical ring on the mesh.

Table 2-55. Unit Masks for TxR_VERT_CYCLES_FULL1

Extension	umask [15:8]	Description
AKC_AG0	bxxxxxxx1	AKC - Agent 0 Ring transactions from Agent 0 destined for the AD ring. Some example include outbound requests, snoop requests, and snoop responses.
AKC_AG1	bxxxxxx1x	AKC - Agent 1 Ring transactions from Agent 0 destined for the AK ring. This is commonly used for credit returns and GO responses.

TxR_VERT_CYCLES_NE0

- **Title:**
- **Category:** Vertical Egress Events
- **Event Code:** 0x96
- **Register Restrictions :**
- **Definition:** Number of cycles the common mesh stop egress was not empty. The egress is used to queue up requests destined for the vertical ring on the mesh.

Table 2-56. Unit Masks for TxR_VERT_CYCLES_NE0 (Sheet 1 of 2)

Extension	umask [15:8]	Description
AD_AG0	bxxxxxxx1	AD - Agent 0 Ring transactions from Agent 0 destined for the AD ring. Some example include outbound requests, snoop requests, and snoop responses.
AK_AG0	bxxxxxx1x	AK - Agent 0 Ring transactions from Agent 0 destined for the AK ring. This is commonly used for credit returns and GO responses.

Table 2-56. Unit Masks for TxR_VERT_CYCLES_NE0 (Sheet 2 of 2)

Extension	umask [15:8]	Description
BL_AG0	bxxxxx1xx	BL - Agent 0 Ring transactions from Agent 0 destined for the BL ring. This is commonly used to send data from the cache to various destinations.
IV_AG0	bxxxx1xxx	IV - Agent 0 Ring transactions from Agent 0 destined for the IV ring. This is commonly used for snoops to the cores.
AD_AG1	bxxx1xxxx	AD - Agent 1 Ring transactions from Agent 1 destined for the AD ring. This is commonly used for outbound requests.
AK_AG1	bxx1xxxxx	AK - Agent 1 Ring transactions from Agent 1 destined for the AK ring.
BL_AG1	bx1xxxxxx	BL - Agent 1 Ring transactions from Agent 1 destined for the BL ring. This is commonly used for transferring write back data to the cache.

TxR_VERT_CYCLES_NE1

- **Title:**
- **Category:** Vertical Egress Events
- **Event Code:** 0x97
- **Register Restrictions :**
- **Definition:** Number of cycles the common mesh stop egress was not empty. The egress is used to queue up requests destined for the vertical ring on the mesh.

Table 2-57. Unit Masks for TxR_VERT_CYCLES_NE1

Extension	umask [15:8]	Description
AKC_AG0	bxxxxxxx1	AKC - Agent 0 Ring transactions from Agent 0 destined for the AD ring. Some example include outbound requests, snoop requests, and snoop responses.
AKC_AG1	bxxxxxx1x	AKC - Agent 1 Ring transactions from Agent 0 destined for the AK ring. This is commonly used for credit returns and GO responses.

TxR_VERT_INSERTS0

- **Title:**
- **Category:** Vertical Egress Events
- **Event Code:** 0x92
- **Register Restrictions :**
- **Definition:** Number of allocations into the common mesh stop egress. The egress is used to queue up requests destined for the vertical ring on the mesh.

Table 2-58. Unit Masks for TxR_VERT_INSERTS0

Extension	umask [15:8]	Description
AD_AG0	bxxxxxxx1	AD - Agent 0 Ring transactions from Agent 0 destined for the AD ring. Some example include outbound requests, snoop requests, and snoop responses.
AK_AG0	bxxxxxx1x	AK - Agent 0 Ring transactions from Agent 0 destined for the AK ring. This is commonly used for credit returns and GO responses.
BL_AG0	bxxxxx1xx	BL - Agent 0 Ring transactions from Agent 0 destined for the BL ring. This is commonly used to send data from the cache to various destinations.
IV_AG0	bxxxx1xxx	IV - Agent 0 Ring transactions from Agent 0 destined for the IV ring. This is commonly used for snoops to the cores.
AD_AG1	bxxx1xxxx	AD - Agent 1 Ring transactions from Agent 1 destined for the AD ring. This is commonly used for outbound requests.
AK_AG1	bxx1xxxxx	AK - Agent 1 Ring transactions from Agent 1 destined for the AK ring.
BL_AG1	bx1xxxxxx	BL - Agent 1 Ring transactions from Agent 1 destined for the BL ring. This is commonly used for transferring write back data to the cache.

TxR_VERT_INSERTS1

- **Title:**
- **Category:** Vertical Egress Events
- **Event Code:** 0x93
- **Register Restrictions :**
- **Definition:** Number of allocations into the common mesh stop egress. The egress is used to queue up requests destined for the vertical ring on the mesh.

Table 2-59. Unit Masks for TxR_VERT_INSERTS1

Extension	umask [15:8]	Description
AKC_AG0	bxxxxxxx1	AKC - Agent 0 Ring transactions from Agent 0 destined for the AD ring. Some example include outbound requests, snoop requests, and snoop responses.
AKC_AG1	bxxxxxx1x	AKC - Agent 1 Ring transactions from Agent 0 destined for the AK ring. This is commonly used for credit returns and GO responses.

TxR_VERT_NACK0

- **Title:**
- **Category:** Vertical Egress Events
- **Event Code:** 0x98
- **Register Restrictions :**

- **Definition:** Counts number of egress packets NACKed onto the vertical ring.

Table 2-60. Unit Masks for TxR_VERT_NACK0

Extension	umask [15:8]	Description
AD_AG0	bxxxxxxx1	AD - Agent 0
AK_AG0	bxxxxxx1x	AK - Agent 0
BL_AG0	bxxxxx1xx	BL - Agent 0
IV_AG0	bxxxx1xxx	IV
AD_AG1	bxxx1xxxx	AD - Agent 1
AK_AG1	bxx1xxxxx	AK - Agent 1
BL_AG1	bx1xxxxxx	BL - Agent 1

TxR_VERT_NACK1

- **Title:**
- **Category:** Vertical Egress Events
- **Event Code:** 0x99
- **Register Restrictions :**
- **Definition:** Counts number of egress packets NACKed onto the vertical ring.

Table 2-61. Unit Masks for TxR_VERT_NACK1

Extension	umask [15:8]	Description
AKC_AG0	bxxxxxxx1	AKC - Agent 0
AKC_AG1	bxxxxxx1x	AKC - Agent 1

TxR_VERT_OCCUPANCY0

- **Title:**
- **Category:** Vertical Egress Events
- **Event Code:** 0x90
- **Register Restrictions :**
- **Definition:** Occupancy event for the egress buffers in the common mesh stop. The egress is used to queue up requests destined for the vertical ring on the mesh.

Table 2-62. Unit Masks for TxR_VERT_OCCUPANCY0

Extension	umask [15:8]	Description
AD_AG0	bxxxxxxx1	AD - Agent 0 Ring transactions from Agent 0 destined for the AD ring. Some example include outbound requests, snoop requests, and snoop responses.
AK_AG0	bxxxxxx1x	AK - Agent 0 Ring transactions from Agent 0 destined for the AK ring. This is commonly used for credit returns and GO responses.
BL_AG0	bxxxxx1xx	BL - Agent 0 Ring transactions from Agent 0 destined for the BL ring. This is commonly used to send data from the cache to various destinations.
IV_AG0	bxxxx1xxx	IV - Agent 0 Ring transactions from Agent 0 destined for the IV ring. This is commonly used for snoops to the cores.
AD_AG1	bxxx1xxxx	AD - Agent 1 Ring transactions from Agent 1 destined for the AD ring. This is commonly used for outbound requests.
AK_AG1	bxx1xxxxx	AK - Agent 1 Ring transactions from Agent 1 destined for the AK ring.
BL_AG1	bx1xxxxxx	BL - Agent 1 Ring transactions from Agent 1 destined for the BL ring. This is commonly used for transferring write back data to the cache.

TxR_VERT_OCCUPANCY1

- **Title:**
- **Category:** Vertical Egress Events
- **Event Code:** 0x91
- **Register Restrictions :**
- **Definition:** Occupancy event for the egress buffers in the common mesh stop. The egress is used to queue up requests destined for the vertical ring on the mesh.

Table 2-63. Unit Masks for TxR_VERT_OCCUPANCY1

Extension	umask [15:8]	Description
AKC_AG0	bxxxxxxx1	AKC - Agent 0 Ring transactions from Agent 0 destined for the AD ring. Some example include outbound requests, snoop requests, and snoop responses.
AKC_AG1	bxxxxxx1x	AKC - Agent 1 Ring transactions from Agent 0 destined for the AK ring. This is commonly used for credit returns and GO responses.

TxR_VERT_STARVED0

- **Title:**
- **Category:** Vertical Egress Events
- **Event Code:** 0x9A
- **Register Restrictions :**

- **Definition:** Counts injection starvation. This starvation is triggered when the CMS egress cannot send a transaction onto the vertical ring for a long period of time.

Table 2-64. Unit Masks for TxR_VERT_STARVED0

Extension	umask [15:8]	Description
AD_AG0	bxxxxxxx1	AD - Agent 0
AK_AG0	bxxxxxx1x	AK - Agent 0
BL_AG0	bxxxxx1xx	BL - Agent 0
IV_AG0	bxxxx1xxx	IV
AD_AG1	bxxx1xxxx	AD - Agent 1
AK_AG1	bxx1xxxxx	AK - Agent 1
BL_AG1	bx1xxxxxx	BL - Agent 1

TxR_VERT_STARVED1

- **Title:**
- **Category:** Vertical Egress Events
- **Event Code:** 0x9B
- **Register Restrictions :**
- **Definition:** Counts injection starvation. This starvation is triggered when the CMS egress cannot send a transaction onto the vertical ring for a long period of time.

Table 2-65. Unit Masks for TxR_VERT_STARVED1

Extension	umask [15:8]	Description
AKC_AG0	bxxxxxxx1	AKC - Agent 0
AKC_AG1	bxxxxxx1x	AKC - Agent 1
TGC	bxxxxx1xx	AKC - Agent 0

VERT_RING_AD_IN_USE

- **Title:**
- **Category:** Vertical In Use Ring Events
- **Event Code:** 0xB0
- **Register Restrictions :**
- **Definition:** Counts the number of cycles that the vertical AD ring is being used at this ring stop. This includes when packets are passing by and when packets are being sunk, but does not include when packets are being sent from the ring stop. We really have two rings -- a clockwise ring and a counter-clockwise ring. On the left side of the ring, the "UP" direction is on the clockwise ring and "DN" is on the counter-clockwise ring. On the right side of the ring, this is reversed. The first half of the CBos are on the left side of the ring, and the second half are on the right side of the ring.

In other words (for example), in a 4c part, Cbo 0 UP AD is NOT the same ring as CBo 2 UP AD because they are on opposite sides of the ring.

Table 2-66. Unit Masks for VERT_RING_AD_IN_USE

Extension	umask [15:8]	Description
UP_EVEN	bxxxxxxx1	Up and Even
UP_ODD	bxxxxxx1x	Up and Odd
DN_EVEN	bxxxxx1xx	Down and Even
DN_ODD	bxxxx1xxx	Down and Odd

VERT_RING_AKC_IN_USE

- **Title:**
- **Category:** Vertical In Use Ring Events
- **Event Code:** 0xB4
- **Register Restrictions :**
- **Definition:** Counts the number of cycles that the vertical AKC ring is being used at this ring stop. This includes when packets are passing by and when packets are being sunk, but does not include when packets are being sent from the ring stop. We really have two rings in JKT -- a clockwise ring and a counter-clockwise ring. On the left side of the ring, the “UP” direction is on the clockwise ring and “DN” is on the counter-clockwise ring. On the right side of the ring, this is reversed. The first half of the CBos are on the left side of the ring, and the second half are on the right side of the ring. In other words (for example), in a 4c part, Cbo 0 UP AD is NOT the same ring as CBo 2 UP AD because they are on opposite sides of the ring.

Table 2-67. Unit Masks for VERT_RING_AKC_IN_USE

Extension	umask [15:8]	Description
UP_EVEN	bxxxxxxx1	Up and Even
UP_ODD	bxxxxxx1x	Up and Odd
DN_EVEN	bxxxxx1xx	Down and Even
DN_ODD	bxxxx1xxx	Down and Odd

VERT_RING_AK_IN_USE

- **Title:**
- **Category:** Vertical In Use Ring Events
- **Event Code:** 0xB1
- **Register Restrictions :**
- **Definition:** Counts the number of cycles that the vertical AK ring is being used at this ring stop. This includes when packets are passing by and when packets are being sunk, but does not include when packets are being sent from the ring stop. We really have two rings in -- a clockwise ring and a counter-clockwise ring. On the left side of the ring, the “UP” direction is on the clockwise ring and “DN” is on the counter-clockwise ring. On the right side of the ring, this is reversed. The first half of the CBos

are on the left side of the ring, and the second half are on the right side of the ring. In other words (for example), in a 4c part, Cbo 0 UP AD is NOT the same ring as CBo 2 UP AD because they are on opposite sides of the ring.

Table 2-68. Unit Masks for VERT_RING_AK_IN_USE

Extension	umask [15:8]	Description
UP_EVEN	bxxxxxxx1	Up and Even
UP_ODD	bxxxxxx1x	Up and Odd
DN_EVEN	bxxxxx1xx	Down and Even
DN_ODD	bxxxx1xxx	Down and Odd

VERT_RING_BL_IN_USE

- **Title:**
- **Category:** Vertical In Use Ring Events
- **Event Code:** 0xB2
- **Register Restrictions :**
- **Definition:** Counts the number of cycles that the vertical BL ring is being used at this ring stop. This includes when packets are passing by and when packets are being sunk, but does not include when packets are being sent from the ring stop. We really have two rings -- a clockwise ring and a counter-clockwise ring. On the left side of the ring, the "UP" direction is on the clockwise ring and "DN" is on the counter-clockwise ring. On the right side of the ring, this is reversed. The first half of the CBos are on the left side of the ring, and the second half are on the right side of the ring. In other words (for example), in a 4c part, Cbo 0 UP AD is NOT the same ring as CBo 2 UP AD because they are on opposite sides of the ring.

Table 2-69. Unit Masks for VERT_RING_BL_IN_USE

Extension	umask [15:8]	Description
UP_EVEN	bxxxxxxx1	Up and Even
UP_ODD	bxxxxxx1x	Up and Odd
DN_EVEN	bxxxxx1xx	Down and Even
DN_ODD	bxxxx1xxx	Down and Odd

VERT_RING_IV_IN_USE

- **Title:**
- **Category:** Vertical In Use Ring Events
- **Event Code:** 0xB3
- **Register Restrictions :**
- **Definition:** Counts the number of cycles that the vertical IV ring is being used at this ring stop. This includes when packets are passing by and when packets are being sunk, but does not include when packets are being sent from the ring stop. There is only 1 IV ring. Therefore, if one wants to monitor the "even" ring, they should select

both UP_EVEN and DN_EVEN. To monitor the "odd" ring, they should select both UP_ODD and DN_ODD.

Table 2-70. Unit Masks for VERT_RING_IV_IN_USE

Extension	umask [15:8]	Description
UP	bxxxxxxx1	Up
DN	bxxxxx1xx	Down

VERT_RING_TGC_IN_USE

- **Title:**
- **Category:** Vertical In Use Ring Events
- **Event Code:** 0xB5
- **Register Restrictions :**
- **Definition:** Counts the number of cycles that the vertical TGC ring is being used at this ring stop. This includes when packets are passing by and when packets are being sunk, but does not include when packets are being sent from the ring stop. We really have two rings in JKT -- a clockwise ring and a counter-clockwise ring. On the left side of the ring, the "UP" direction is on the clockwise ring and "DN" is on the counter-clockwise ring. On the right side of the ring, this is reversed. The first half of the CBos are on the left side of the ring, and the second half are on the right side of the ring. In other words (for example), in a 4c part, Cbo 0 UP AD is NOT the same ring as CBo 2 UP AD because they are on opposite sides of the ring.

Table 2-71. Unit Masks for VERT_RING_TGC_IN_USE

Extension	umask [15:8]	Description
UP_EVEN	bxxxxxxx1	Up and Even
UP_ODD	bxxxxxx1x	Up and Odd
DN_EVEN	bxxxxx1xx	Down and Even
DN_ODD	bxxxx1xxx	Down and Odd

2.2 Caching/Home Agent (CHA) Performance Monitoring

The LLC Coherence engine and Home agent (CHA) merges the caching agent and Home Agent (HA) responsibilities of the chip into a single block. In its capacity as a caching agent, the CHA manages the interface between the core the IIO devices and the Last Level Cache (LLC). In its capacity as a home agent, the CHA manages the interface between the LLC and the rest of the Intel UPI coherent fabric as well as the on die memory controller.

All core and IIO DMA transactions that access the LLC are directed from their source to a CHA via the mesh interconnect. The CHA is responsible for managing data delivery from the LLC to the requester and maintaining coherence between the all the cores and

IIO devices within the socket that share the LLC. It is also responsible for generating snoops and collecting snoop responses from the local cores when the MESIF protocol requires it.

Similarly, all incoming traffic from remote sockets that maps to the socket's local memory are directed from the Intel UPI link(s) to a CHA via the mesh interconnect. The CHA is responsible for managing the coherence across all sockets in the system for the socket's memory following the protocols defined in the Intel UPI Specification. It manages directory state for the local memory, conflicts, and memory ordering rules for such requests.

In the process of maintaining cache coherency within the socket, and across the system in a multi-socket system, the CHA is the gate keeper for all Intel® UPI interconnect messages that have addresses mapping to the socket's memory as well as the originator of all Intel® UPI interconnect messages the originate from cores within its socket when attempts are made to access memory in another socket. It is responsible for ensuring that all Intel® UPI messages that pass through the socket remain coherent.

The CHA can manage a large number of simultaneous requests in parallel, but in order to maintain proper memory ordering it does ensure that whenever multiple incoming requests to the same address are pending (whether they originated from a core or IIO device within the socket or came in from another socket through one of the Intel UPI links) only one of those requests is being processed at a time. Considering this LLC cache is not inclusive of the IA cores' internal caches, the total cache capacity of the socket is much larger than the LLC capacity and each CHA is responsible for monitor a portion of that available IA core cache capacity for the purpose of maintaining coherence between the IA core caches and the rest of the Intel UPI coherent fabric.

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

2.2.1 CHA Performance Monitoring Overview

Each of the CHAs in the 5th Gen Intel® Xeon® Scalable Processor's uncore supports event monitoring through four 48-bit wide counters (Cn_MSR_PMON_CTR{3:0}). With but a small number of exceptions, each of these counters can be programmed (Cn_MSR_PMON_CTL{3:0}) for any available event.

Some uncore performance events that monitor transaction activities require additional details that must be programmed in a filter register. Each CHA provides an additional filter register and allows only one such event to be programmed at a given time, see Section 1.3.2.1, "Global PMON Global Control/Status Registers".

2.2.1.1 Special Note on CHA Occupancy Events

Although only counter 0 supports occupancy events, it is possible to program counters 1-3 to monitor the same occupancy event by selecting the "OCCUPANCY_COUNTER0" event code on counters 1-3.

This allows:

- Thresholding on all four counters.
  While no more than one queue can be monitored at a time, it is possible to setup different queue occupancy thresholds on each of the four counters. For example, if one wanted to monitor the IRQ, one could setup thresholds of 1, 7, 14, and 18 to get a picture of the time spent at different occupancies in the IRQ.
- Average Latency and Average Occupancy.
  It can be useful to monitor the average occupancy in a queue as well as the average number of items in the queue. One could program counter 0 to accumulate the occupancy, counter 1 with the queue's allocations event, and counter 2 with the OCCUPANCY_COUNTER0 event and a threshold of 1. Latency could then be calculated by counter 0 / counter 1, and occupancy by counter 0 / counter 2.

2.2.2 Additional CHA Performance Monitoring

2.2.2.1 CHA PMON Counter Control - Difference from Baseline

CHA performance monitoring control registers provide a small amount of additional functionality. The following table defines those cases.

Figure 2-1. CHA Counter Control Register for 5th Gen Intel® Xeon® Scalable Processor

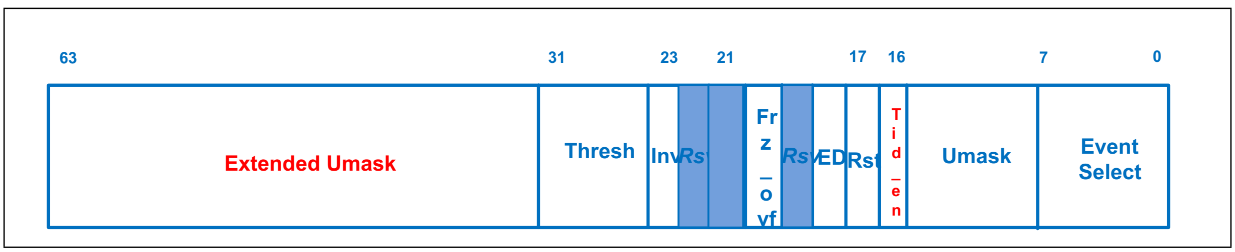


Table 2-72. Cn_MSR_PMON_CTL{3-0} Register – Field Definitions

Field	Bits	Attr	HW Reset Val	Description
tid_en	16	RW-V	0	TID Filter Enable
Extended Umask	32:63		0	Extra Filtering

Figure 2-2. UmaskExt Filter Details for TOR_INSERT/OCCUPANCY Events

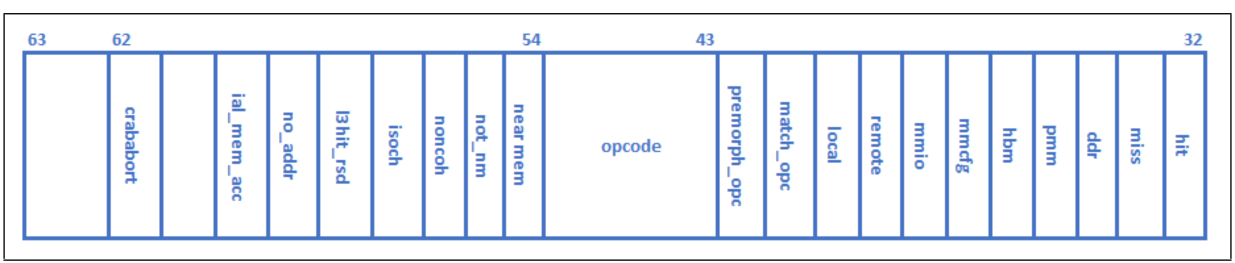


Table 2-73. UmaskExt Filter Details for TOR_INSERT/OCCUPANCY Events

Field	Bits	Atrtr	HW Reset Val	Description
crababort	62	RW	0	
ia_mem_acc	60	RW	0	
no_addr	59	RW	0	
l3hit_rsd	58	RW	0	
isoc	57	RW	0	Match on ISOC Requests
nc	56	RW	0	Match on Non-Coherent Requests
not_nm	55	RW	0	Just Match on Non Near Memory Cacheable Accesses. b55 is XORed with b54. No filtering applied if both bits are either 0 or 1
nm	54	RW	0	Just Match on Near Memory Cacheable Accesses
opc (11b IDI Opcode w/top 2b 0x3)	53:43	RW	0	Match on Opcode (see Section 3.1.1, “Reference for CHA Packet Matching”) Can be used to track transaction by Opocde relevant to each key queue in the CHA pipeline: IPQ, IRQ, RRQ and WBQ
premorph_opc	42	RW	0	Filter by PreMorphed Opcodes
match_opc	41	RW	0	Filter by Opcodes
loc	40	RW	0	Just Match on Local Node Target. b40 is XORed with b39. No filtering applied if both bits are either 0 or 1
rem	39	RW	0	Just Match on Remote Node Target
mmio	38	RW	0	Filter on requests to memory mapped to MMIO space
mmcfg	37	RW	0	Filter on requests to memory mapped to MMCFG space
hbm	36	RW	0	Filter on requests to memory mapped to HBM
pmm	35	RW	0	Filter on requests to memory mapped to PMM
ddr	34	RW	0	Match on Remote Node Target
miss	33	RW	0	Just entries that Missed the LLC. b33 is XORed with b32. No filtering applied if both bits are either 0 or 1
hit	32	RW	0	Just entries that Hit the LLC

Figure 2-3. UmaskExt Filter Details for the LLC_LOOKUP Event

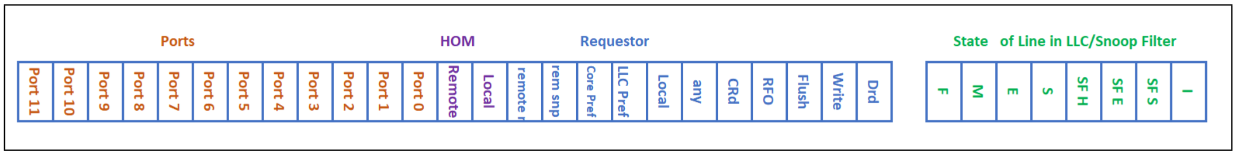


Table 2-74. UmaskExt Filter Details for LLC_LOOKUP Events

Field	Bits	Atrtr	HW Reset Val	Description
IIO_Port11	56	RW	0	M2IOSF Port 11
IIO_Port10	55	RW	0	M2IOSF Port 10
IIO_Port9	54	RW	0	M2IOSF Port 9
IIO_Port8	53	RW	0	M2IOSF Port 8
IIO_Port7	52	RW	0	M2IOSF Port 7
IIO_Port6	51	RW	0	M2IOSF Port 6
IIO_Port5	50	RW	0	M2IOSF Port 5
IIO_Port4	49	RW	0	M2IOSF Port 4
IIO_Port3	48	RW	0	M2IOSF Port 3
IIO_Port2	47	RW	0	M2IOSF Port 2
IIO_Port1	46	RW	0	M2IOSF Port 1
IIO_Port0	45	RW	0	M2IOSF Port 0
remote hom	44	RW	0	Transactions to remotely homed addresses
local hom	43	RW	0	Transactions to locally homed addresses
remote non-snoop	42	RW	0	Non-snoop transactions to the LLC from a remote agent
remote snoop	41	RW	0	Snoop transactions to the LLC from a remote agent
Core prefetch	40	RW	0	Any local prefetch to LLC from Core
LLC prefetch	39	RW	0	Any local prefetch to LLC from an LLC
local	38	RW	0	Any local transaction to LLC, including prefetches from Core
any	37	RW	0	Any local or remote transaction to the LLC. Includes prefetches
CRd	36	RW	0	Code Reads- local or remote. includes prefetches
RFO	35	RW	0	RFOs - local or remote. includes prefetches
flush or inv	34	RW	0	Flush or Invalidates
writes	33	RW	0	All write transactions to the LLC - including write backs to LLC and uncacheable write transactions Does not include evict cleans or invalidates
DRd	32	RW	0	Data Reads- local or remote. includes prefetches
state	umask field 15:8	RW	0	Select state to monitor for **LLC_LOOKUP** event.Setting multiple bits in this field will allow a user to track multiple states. bxx1xxxxxxx - LLC - F state. bxxx1xxxxxx - LLC - M state. bxxxx1xxxxx - LLC - E state. bxxxxx1xxxx - LLC - S state. bxxxxxx1xxx - SF - H state bxxxxxxx1xx - SF - E state. bxxxxxxxx1x - SF - S state. bxxxxxxxxx1 - LLC - I state.

Note: The Request field will be ANDed with State and HOM fields.

2.2.2.2 CHA Filter Registers (Cn_MSR_PMON_BOX_FILTER0)

Any of the CHA events may be filtered by thread/core-ID. To do so, the control register's *.tid_en* bit must be set to 1 and the *tid* field in the FILTER register filled out. Only one of these filtering criteria may be applied at a time.

Figure 2-4. CHA PMON Filter Register

Table 2-75. Cn_MSR_PMON_BOX_FILTER Register – Field Definitions

Field	Bits	Atrtr	HW Reset Val	Description
rsv	31:9	RV	0	Reserved.SW must set to 0 else behavior is undefined
tid	9:0	0	0	[9:3] Core-ID [2:0] Thread 3-0 When *.tid_en* is 0; the specified counter will count ALL events. To filter on a specific logical core, set Core-ID to the desired core number and set the TID field to the desired thread. To filter on a source/destination other than an IA core, set Core-ID to one of the following and set TID to 0.

2.2.3 CHA Performance Monitoring Events

The performance monitoring events within the CHA include all events internal to the LLC and HA as well as events which track mesh related activity at the CHA/core mesh stops (see Section 2.1.1, "Mesh Performance Monitoring Events" for the available Mesh Stop events).

CHA performance monitoring events can be used to track LLC access rates, LLC hit/ miss rates, LLC eviction and fill rates, HA access rates, HA conflicts, and to detect evidence of back pressure on the internal CHA pipelines. In addition, the CHA has performance monitoring events for tracking MESIF state transitions that occur as a result of data sharing across sockets in a multi-socket system.

Every event in the CHA is from the point of view of the CHA and is not associated with any specific core since all cores in the socket send their LLC transactions to all CHAs in the socket.

There are separate sets of counters for each CHA instance. For any event, to get an aggregate count of that event for the entire LLC, the counts across the CHA instances must be added together. The counts can be averaged across the CHA instances to get a view of the typical count of an event from the perspective of the individual CHAs. Individual per-CHA deviations from the average can be used to identify hot-spotting across the CHAs or other evidences of non-uniformity in LLC behavior across the CHAs. 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 CHAs would indicate hot-spotting.

2.2.3.1 Acronyms Frequently Used in CHA Events

The Rings:

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

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

AK (Acknowledge) Ring - Acknowledges Intel UPI to CHA and CHA to Core. Carries snoop responses from Core to CHA.

IV (Invalidate) Ring - CHA Snoop requests of core caches.

2.2.3.2 Key Queues

IRQ - Requests from IA Cores.

IPQ - Ingress Probe Queue on AD Ring. Remote socket snoops sent from Intel UPI LL.

ISMQ - Ingress Subsequent Messages (response queue). Associated with message responses to ingress requests (for example, data responses, Intel UPI completion messages, core snoop response messages and the GO reset queue).

PRQ - Requests from IIO.

RRQ - Remote Request Queue. Remote socket read requests, from Intel UPI to the local home agent.

WBQ - Write back Queue. Remote socket write requests, from Intel UPI to the local home agent.

TOR - Table Of Requests. Tracks pending CHA transactions.

RxC (aka IGR) /TxC (aka EGR) - Ingress, requests from cores (by way of the CMS), and egress, requests headed for the mesh (by way of the CMS), queues.

2.2.4 CHA Box Events Ordered By Code

The following table summarizes the directly measured CHA Box events.

Table 2-76. Measured CHA Box Events (Sheet 1 of 3)

Symbol Name	Event Code	Ctrs	Description
CLOCKTICKS	0x1	0-3	Clock ticks of the uncore caching and home agent (CHA)
RxC_OCCUPANCY	0x11	0	Ingress (from CMS) Occupancy
RxC_INSERTS	0x13	0-3	Ingress (from CMS) Allocations
RxC_IRQ0_REJECT	0x18	0-3	IRQ Requests (from CMS) Rejected - Set 0
RxC_IRQ1_REJECT	0x19	0-3	IRQ Requests (from CMS) Rejected - Set 1
RxC_PRQ0_REJECT	0x20	0-3	PRQ Requests (from CMS) Rejected - Set 0
RxC_PRQ1_REJECT	0x21	0-3	PRQ Requests (from CMS) Rejected - Set 1

Table 2-76. Measured CHA Box Events (Sheet 2 of 3)

Symbol Name	Event Code	Ctrs	Description
RxC_IPQ0_REJECT	0x22	0-3	IPQ Requests (from CMS) Rejected - Set 0
RxC_IPQ1_REJECT	0x23	0-3	IPQ Requests (from CMS) Rejected - Set 1
RxC_ISMQ0_REJECT	0x24	0-3	ISMQ Rejects - Set 0
RxC_ISMQ1_REJECT	0x25	0-3	ISMQ Rejects - Set 1
RxC_RRQ0_REJECT	0x26	0-3	RRQ Rejects - Set 0
RxC_RRQ1_REJECT	0x27	0-3	RRQ Rejects - Set 1
RxC_WBQ0_REJECT	0x28	0-3	WBQ Rejects - Set 0
RxC_WBQ1_REJECT	0x29	0-3	WBQ Rejects - Set 1
RxC_REQ_Q0_RETRY	0x2A	0-3	Request Queue Retries - Set 0
RxC_REQ_Q1_RETRY	0x2B	0-3	Request Queue Retries - Set 1
RxC_ISMQ0_RETRY	0x2C	0-3	ISMQ Retries - Set 0
RxC_ISMQ1_RETRY	0x2D	0-3	ISMQ Retries - Set 1
RxC_OTHER0_RETRY	0x2E	0-3	Other Retries - Set 0
RxC_OTHER1_RETRY	0x2F	0-3	Other Retries - Set 1
LLC_LOOKUP	0x34	0-3	Cache Lookups
TOR_INSERTS	0x35	0-3	TOR Inserts
TOR_OCCUPANCY	0x36	0	TOR Occupancy
LLC_VICTIMS	0x37	0-3	Lines Victimized
MISC	0x39	0-3	Cbo Misc
SF_EVICTION	0x3D	0-3	Snoop Filter Capacity Evictions
REQUESTS	0x50	0-3	HA Read and Write Requests
SNOOPS_SENT	0x51	0-3	Snoops Sent
DIR_LOOKUP	0x53	0-3	Multi-socket cacheline directory state lookups
DIR_UPDATE	0x54	0-3	Multi-socket cacheline directory state updates
OSB	0x55	0-3	OSB Snoop Broadcast
WB_PUSH_MTOI	0x56	0-3	WbPushMtoI
BYPASS_CHA_IMC	0x57	0-3	CHA to iMC Bypass
READ_NO_CREDITS	0x58	0-3	CHA iMC CHNx READ Credits Empty
IMC_READS_COUNT	0x59	0-3	HA to iMC Reads Issued
WRITE_NO_CREDITS	0x5A	0-3	CHA iMC CHNx WRITE Credits Empty
IMC_WRITES_COUNT	0x5B	0-3	CHA to iMC Full Line Writes Issued
SNOOP_RESP	0x5C	0-3	Snoop Responses Received
SNOOP_RESP_LOCAL	0x5D	0-3	Snoop Responses Received Local
HITME_LOOKUP	0x5E	0-3	Counts Number of times HitMe Cache is accessed
HITME_HIT	0x5F	0-3	Counts Number of Hits in HitMe Cache
HITME_MISS	0x60	0-3	Counts Number of Misses in HitMe Cache
HITME_UPDATE	0x61	0-3	Counts the number of Allocate/Update to HitMe Cache
PMM_MEMMODE_NM_SETCONFLICTS	0x64	0-3	PMM Memory Mode related events

Table 2-76. Measured CHA Box Events (Sheet 3 of 3)

Symbol Name	Event Code	Ctrs	Description
PMM_MEMMODE_NM_INVITOX	0x65	0-3	
PMM_QOS	0x66	0-3	
PMM_QOS_OCCUPANCY	0x67	0-3	
MISC2	0x6A	0-3	Cbo Misc2
DIRECT_GO_OPC	0x6D	0-3	Direct GO
DIRECT_GO	0x6E	0-3	Direct GO

2.2.5 CHA Box Common Metrics (Derived Events)

The following table summarizes metrics commonly calculated from CHA Box events

Table 2-77. Common Metrics from CHA Box Events Calculations (Sheet 1 of 3)

Symbol Name: Definition	Equation
AVG_CRD_MISS_LATENCY: Average Latency of Code Reads from an iA Core that miss the LLC	(TOR_OCCUPANCY.IA_MISS_CRD + TOR_OCCUPANCY.IA_MISS_CRD_PREF) / (TOR_INSERTS.IA_MISS_CRD + TOR_INSERTS.IA_MISS_CRD_PREF)
AVG_DEMAND_RD_HIT_LATENCY: Average Latency of Data Reads that hit the LLC	TOR_OCCUPANCY.IA_HIT_DRD / TOR_INSERTS.IA_HIT_DRD
AVG_DEMAND_RD_MISS_LOCAL_LATENCY: Average Latency of Data Reads from an IA Core that miss the LLC and were satsified by Local Memory	TOR_OCCUPANCY.IA_MISS_DRD_LOCAL / TOR_INSERTS.IA_MISS_DRD_LOCAL
AVG_DEMAND_RD_MISS_REMOTE_LATENCY: Average Latency of Data Reads from an iA Core that miss the LLC and were satsified by a Remote cache or Remote Memory	TOR_OCCUPANCY.IA_MISS_DRD_REMOTE / TOR_INSERTS.IA_MISS_DRD_REMOTE
AVG_DRD_MISS_LATENCY: Average Latency of Data Reads or Data Read Prefetches from an IA Core that miss the LLC	(TOR_OCCUPANCY.IA_MISS_DRD + TOR_OCCUPANCY.IA_MISS_DRD_PREF) / (TOR_INSERTS.IA_MISS_DRD + TOR_INSERTS.IA_MISS_DRD_PREF)
AVG_IA_CRD_LLC_HIT_LATENCY: Average Latency of Code Reads from an iA Core that miss the LLC	TOR_OCCUPANCY.IA_HIT_CRD / TOR_INSERTS.IA_HIT_CRD
AVG_INGRESS_DEPTH: Average Depth of the Ingress Queue through the sample interval	RxC_OCCUPANCY.IRQ / SAMPLE_INTERVAL
AVG_INGRESS_LATENCY: Average Latency of Requests through the Ingress Queue in Uncore Clocks	RxC_OCCUPANCY.IRQ / RxC_INSERTS.IRQ
AVG_INGRESS_LATENCY_WHEN_NE: Average Latency of Requests through the Ingress Queue in Uncore Clocks when Ingress Queue has at least one entry	RxC_OCCUPANCY.IRQ / COUNTER0_OCCUPANCY{edge_det,thresh=0x1}
AVG_RFO_MISS_LATENCY: Average Latency of RFOs from an iA Core that miss the LLC	(TOR_OCCUPANCY.IA_MISS_RFO + TOR_OCCUPANCY.IA_MISS_RFO_PREF) / (TOR_INSERTS.IA_MISS_RFO + TOR_INSERTS.IA_MISS_RFO_PREF)
AVG_TOR_DRDS_MISS_WHEN_NE: Average Number of Data Read Entries that Miss the LLC when the TOR is not empty.	TOR_OCCUPANCY.IA_MISS_DRD / COUNTER0_OCCUPANCY{edge_det,thresh=0x1}

Table 2-77. Common Metrics from CHA Box Events Calculations (Sheet 2 of 3)

Symbol Name: Definition	Equation
AVG_TOR_DRDS_WHEN_NE: Average Number of Data Read Entries when the TOR is not empty.	TOR_OCCUPANCY.IA_DRD / COUNTER0_OCCUPANCY{edge_det,thresh=0x1}
CYC_INGRESS_BLOCKED: Cycles the Ingress Request Queue arbiter was Blocked	RxC_EXT_STARVED.IRQ / SAMPLE_INTERVAL
FAST_STR_LLC_HIT: Number of ItoM (fast string) operations that reference the LLC	TOR_INSERTS.IA_HIT_ITOM
FAST_STR_LLC_MISS: Number of ItoM (fast string) operations that miss the LLC	TOR_INSERTS.IA_MISS_ITOM
INGRESS_REJ_V_INS: Ratio of Ingress Request Entries that were rejected vs. inserted	RxC_INSERTS.IRQ_REJECTED / RxC_INSERTS.IRQ
LLC_CRD_MISS_TO_LOC_MEM: LLC Code Read and Code Prefetch misses satisfied by local memory.	TOR_INSERTS.IA_MISS_CRD_PREF_LOCAL + TOR_INSERTS.IA_MISS_CRD_LOCAL
LLC_CRD_MISS_TO_REM_MEM: LLC Code Read and Code Read Prefetch misses satisfied by a remote cache or remote memory.	TOR_INSERTS.IA_MISS_CRD_PREF_REMOTE + TOR_INSERTS.IA_MISS_CRD_REMOTE
LLC_DRD_MISS_PCT:	LLC_LOOKUP.DATA_READ_MISS / LLC_LOOKUP.DATA_READ_ALL
LLC_DRD_MISS_TO_LOC_MEM: LLC Data Read and Data Prefetch misses satisfied by local memory.	TOR_INSERTS.IA_MISS_DRD_LOCAL
LLC_DRD_MISS_TO_REM_MEM: LLC Data Read and Data Prefetch misses satisfied by a remote cache or remote memory.	TOR_INSERTS.IA_MISS_DRD_REMOTE
LLC_DRD_PREFETCH_HITS:	TOR_INSERTS.IA_HIT_DRD_PREF
LLC_DRD_PREFETCH_MISSES:	TOR_INSERTS.IA_MISS_DRD_PREF
LLC_IA_CRD_HITS:	TOR_INSERTS.IA_HIT_CRD
LLC_MPI: LLC Misses Per Instruction (code, read, RFO and prefetches)	LLC_LOOKUP.MISS_ALL / INST_RETIRED.ALL (on Core)
LLC_PCIE_DATA_BYTES: LLC write miss (disk/network reads) bandwidth in MB	TOR_INSERTS.IO_ITOM * 64
LLC_RFO_MISS_PCT: LLC RFO Miss Ratio	TOR_INSERTS.IA_MISS_RFO / TOR_INSERTS.IA_RFO
LLC_RFO_MISS_TO_LOC_MEM: LLC RFO and RFO Prefetch misses satisfied by local memory.	TOR_INSERTS.IA_MISS_RFO_LOCAL
LLC_RFO_MISS_TO_REM_MEM: LLC RFO and RFO Prefetch misses satisfied by a remote cache or remote memory.	TOR_INSERTS.IA_MISS_RFO_REMOTE
LLC_RFO_PREFETCH_HITS:	TOR_INSERTS.IA_HIT_RFO_PREF
LLC_RFO_PREFETCH_MISSES:	TOR_INSERTS.IA_MISS_RFO_PREF
MEM_WB_BYTES: Data written back to memory in Number of Bytes	LLC_VICTIMS.M_STATE * 64
MMIO_READ_BW: IO Read Bandwidth in MB - Disk or Network Reads	TOR_INSERTS.IA_MISS_UCRDF * 64 / 1000000

Table 2-77. Common Metrics from CHA Box Events Calculations (Sheet 3 of 3)

Symbol Name: Definition	Equation
MMIO_WRITE_BW: IO Write Bandwidth in MB - Disk or Network Writes	TOR_INSERTS.IA_MISS_WIL* 64 / 1000000
PCIE_FULL_WRITES: Number of full PCI writes	TOR_INSERTS.IO_ITOM
PCI_PARTIAL_WRITES: Number of partial PCI writes	TOR_INSERTS.IO_RFO
PCI_READS: Number of PCI reads	TOR_INSERTS.IO_PCIRDCUR
PCT_RD_REQUESTS: Percentage of HA traffic that is from Read Requests	REQUESTS.READS / (REQUESTS.READS + REQUESTS.WRITES)
PCT_WR_REQUESTS: Percentage of HA traffic that is from Write Requests	REQUESTS.WRITES / (REQUESTS.READS + REQUESTS.WRITES)
STREAMED_FULL_STORES:	TOR_INSERTS.IA_WCILF
STREAMED_FULL_STORES.MISS_LOCAL_TO_DDR:	TOR_INSERTS.IA_MISS_LOCAL_WCILF_DDR
STREAMED_FULL_STORES.MISS_REMOTE_TO_DDR:	TOR_INSERTS.IA_MISS_REMOTE_WCILF_DDR
STREAMED_FULL_STORES.MISS_TO_DDR:	TOR_INSERTS.IA_MISS_WCILF_DDR
STREAMED_PART_STORES:	TOR_INSERTS.IA_WCIL
STREAMED_PART_STORES.MISS_LOCAL_TO_DDR:	TOR_INSERTS.IA_MISS_LOCAL_WCIL_DDR
STREAMED_PART_STORES.MISS_REMOTE_TO_DDR:	TOR_INSERTS.IA_MISS_REMOTE_WCIL_DDR
STREAMED_PART_STORES.MISS_TO_DDR:	TOR_INSERTS.IA_MISS_WCIL_DDR

2.2.6 CHA Box Performance Monitor Event List

The section enumerates 5th Gen Intel® Xeon® Scalable Processor performance monitoring events for the CHA Box.

BYPASS_CHA_IMC

- **Title:**
- **Category:** HA Bypass Events
- **Event Code:** 0x57
- **Register Restrictions :** 0-3
- **Definition:** Counts the number of times when the CHA was able to bypass HA pipe on the way to iMC. This is a latency optimization for situations when there is light loadings on the memory subsystem. This can be filtered by when the bypass was taken and when it was not.

Table 2-78. Unit Masks for BYPASS_CHA_IMC

Extension	umask [15:8]	Description
TAKEN	bxxxxxxx1	Taken Filter for transactions that succeeded in taking the full bypass.
INTERMEDIATE	bxxxxxx1x	Intermediate bypass Taken Filter for transactions that succeeded in taking the intermediate bypass.
NOT_TAKEN	bxxxxx1xx	Not Taken Filter for transactions that could not take the bypass, and issues a read to memory. Note that transactions that did not take the bypass but did not issue read to memory will not be counted.

CLOCKTICKS

- **Title:**
- **Category:** Clocktick Events
- **Event Code:** 0x1
- **Register Restrictions :** 0-3
- **Definition:**

DIRECT_GO

- **Title:**
- **Category:** DIRECT GO Events
- **Event Code:** 0x6E
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-79. Unit Masks for DIRECT_GO

Extension	umask [15:8]	Description
HA_TOR_DEALLOC	bxxxxxxx1	
HA_SUPPRESS_NO_D2C	bxxxxxx1x	
HA_SUPPRESS_DRD	bxxxxx1xx	

DIRECT_GO_OPC

- **Title:**
- **Category:** DIRECT GO Events
- **Event Code:** 0x6D
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-80. Unit Masks for DIRECT_GO_OPC

Extension	umask [15:8]	Description
EXTCMP	bxxxxxxx1	
PULL	bxxxxxx1x	
GO	bxxxxx1xx	
GO_PULL	bxxxx1xxx	
FAST_GO	bxxx1xxxx	
FAST_GO_PULL	bxx1xxxxx	
NOP	bx1xxxxxx	
IDLE_DUE_SUPPRESS	b1xxxxxxx	

DIR_LOOKUP

- **Title:**
- **Category:** HA Directory Events
- **Event Code:** 0x53
- **Register Restrictions :** 0-3
- **Definition:** Counts the number of transactions that looked up the directory. Can be filtered by requests that had to snoop and those that did not have to.

Table 2-81. Unit Masks for DIR_LOOKUP

Extension	umask [15:8]	Description
SNP	bxxxxxxx1	Snoop Needed Filters for transactions that had to send one or more snoops because the directory was not clean.
NO_SNP	bxxxxxx1x	Snoop Not Needed Filters for transactions that did not have to send any snoops because the directory was clean.

DIR_UPDATE

- **Title:**
- **Category:** HA Directory Events
- **Event Code:** 0x54
- **Register Restrictions :** 0-3
- **Definition:** Counts the number of directory updates that were required. These result in writes to the memory controller. This can be filtered by directory sets and directory clears.

Table 2-82. Unit Masks for DIR_UPDATE

Extension	umask [15:8]	Description
HA	bxxxxxxx1	Directory Updated memory write from HA pipe Counts only directory update Memory writes issued from the HA pipe. Note that any directory update which are part of EWB or IWB are not counted.
TOR	bxxxxxx1x	Directory Updated memory write from TOR pipe Counts only directory update Memory writes issued from the TOR pipe which are the result of remote transaction hitting the SF/LLC and returning data C2C. Note that any directory update which are part of EWB or IWB are not counted.

HITME_HIT

- **Title:**
- **Category:** HA HitME Events
- **Event Code:** 0x5F
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-83. Unit Masks for HITME_HIT

Extension	umask [15:8]	Description
EX_RDS	bxxxxxxx1	Remote socket read requests that hit in E state
SHARED_NONOWNREQ	bxxxxxx1x	Remote socket non-ownership read requests that hit in S state
SHARED_OWNREQ	bxxxxx1xx	Remote socket ownership read requests that hit in S state
WBMTOE	bxxxx1xxx	Remote socket WBMtoE requests
WBMTOI_OR_S	bxxx1xxxx	Remote socket write back to I or S requests

HITME_LOOKUP

- **Title:**
- **Category:** HA HitME Events
- **Event Code:** 0x5E
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-84. Unit Masks for HITME_LOOKUP

Extension	umask [15:8]	Description
READ	bxxxxxxx1	Remote socket read requests
WRITE	bxxxxxx1x	Remote socket write (that is, write back) requests

HITME_MISS

- **Title:**
- **Category:** HA HitME Events
- **Event Code:** 0x60
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-85. Unit Masks for HITME_MISS

Extension	umask [15:8]	Description
SHARED_RDINVOWN	bxx1xxxxx	Remote socket RdInvOwn requests to shared line SF/LLC HitS/F and op is RdInvOwn
NOTSHARED_RDINVOWN	bx1xxxxxx	Remote socket RdInvOwn requests that are not to shared line No SF/LLC HitS/F and op is RdInvOwn
READ_OR_INV	b1xxxxxxx	Remote socket read or invalidate requests op is RdCode, RdData, RdDataMigratory, RdCur, RdInv, Inv*

HITME_UPDATE

- **Title:**
- **Category:** HA HitME Pipe Events
- **Event Code:** 0x61
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-86. Unit Masks for HITME_UPDATE

Extension	umask [15:8]	Description
DEALLOCATE_RSPFWDI_LOC	bxxxxxxx1	Op is RspIFwd or RspIFwdWb for a local request Received RspFwdI* for a local request, but converted HitME SF entry
RSPFWDI_REM	bxxxxxx1x	Op is RspIFwd or RspIFwdWb for a remote request Updated HitME RspFwdI* or local HitM/E received for a remote request
SHARED	bxxxxx1xx	Update HitMe Cache to SHARed
RDINVOWN	bxxxx1xxx	Update HitMe Cache on RdInvOwn even if not RspFwdI*
DEALLOCATE	bxxx1xxxx	Deallocate HtiME Reads without RspFwdI*

IMC_READS_COUNT

- **Title:**
- **Category:** MC Credit and Traffic Events
- **Event Code:** 0x59
- **Register Restrictions :** 0-3
- **Definition:** Count of the number of reads issued to any of the memory controller channels. This can be filtered by the priority of the reads.

Table 2-87. Unit Masks for IMC_READS_COUNT

Extension	umask [15:8]	Description
NORMAL	bxxxxxxx1	Normal
PRIORITY	bxxxxxx1x	ISOCH

IMC_WRITES_COUNT

- **Title:**
- **Category:** MC Credit and Traffic Events
- **Event Code:** 0x5B
- **Register Restrictions :** 0-3
- **Definition:** Counts the total number of full line writes issued from the HA into the memory controller.

Table 2-88. Unit Masks for IMC_WRITES_COUNT

Extension	umask [15:8]	Description
FULL	bxxxxxxx1	Full Line Non-ISOCH
PARTIAL	bxxxxxx1x	Partial Non-ISOCH
FULL_PRIORITY	bxxxxx1xx	ISOCH Full Line
PARTIAL_PRIORITY	bxxxx1xxx	ISOCH Partial

LLC_LOOKUP

- **Title:**
- **Category:** CACHE Events
- **Event Code:** 0x34
- **Register Restrictions :** 0-3
- **Definition:** Counts the number of times the LLC was accessed - this includes code, data, prefetches and hints coming from L2. This has numerous filters available. Note the non-standard filtering equation. This event will count requests that lookup the cache multiple times with multiple increments. One must ALWAYS select a state or states (in the umask field) to match. Otherwise, the event will count nothing.

Table 2-89. Unit Masks for LLC_LOOKUP (Sheet 1 of 4)

Extension	umask [15:8]	xtra [57:32]	Description
ANY_F	bxxxxxxxx	bxxxxxxxxx xxxxxxxxxx 1xxxxx	All Request Filter Any local or remote transaction to the LLC, including prefetch.
CODE_READ_F	bxxxxxxxx	bxxxxxxxxx xxxxxxxxxx x1xxxx	CRd Request Filter Local or remote CRd transactions to the LLC. This includes CRd prefetch.

Table 2-89. Unit Masks for LLC_LOOKUP (Sheet 2 of 4)

Extension	umask [15:8]	xtra [57:32]	Description
COREPREF_LOCAL_F	bxxxxxxxx	bxxxxxxxxx xxxxxxx1xx xxxxxx	Local LLC prefetch requests (from core/L2) Filter Any local prefetch to the LLC from core/L2
COREPREF_OR_DMND_LOCAL_F	bxxxxxxxx	bxxxxxxxxx xxxxxxxxx1 xxxxxx	Local request Filter Any local transaction to the LLC, including prefetches from the Core
DATA_READ_F	bxxxxxxxx	bxxxxxxxxx xxxxxxxxxx xxxxx1	Data Read Request Filter Read transactions.
FLUSH_OR_INV_F	bxxxxxxxx	bxxxxxxxxx xxxxxxxxxx xxx1xx	Flush or Invalidate Filter
IIO_PORT0	bxxxxxxxx	bxxxxxxxxx xx1xxxxxxx xxxxxx	M2IOSF Port0
IIO_PORT1	bxxxxxxxx	bxxxxxxxxx x1xxxxxxxx xxxxxx	M2IOSF Port1
IIO_PORT10	bxxxxxxxx	bx1xxxxxxx xxxxxxxxxx xxxxxx	M2IOSF Port10
IIO_PORT11	bxxxxxxxx	b1xxxxxxxx xxxxxxxxxx xxxxxx	M2IOSF Port11
IIO_PORT2	bxxxxxxxx	bxxxxxxxxx 1xxxxxxxxx xxxxxx	M2IOSF Port2
IIO_PORT3	bxxxxxxxx	bxxxxxxxx1 xxxxxxxxxx xxxxxx	M2IOSF Port3
IIO_PORT4	bxxxxxxxx	bxxxxxxx1x xxxxxxxxxx xxxxxx	M2IOSF Port4
IIO_PORT5	bxxxxxxxx	bxxxxxx1xx xxxxxxxxxx xxxxxx	M2IOSF Port5
IIO_PORT6	bxxxxxxxx	bxxxxx1xxx xxxxxxxxxx xxxxxx	M2IOSF Port6
IIO_PORT7	bxxxxxxxx	bxxxx1xxxx xxxxxxxxxx xxxxxx	M2IOSF Port7
IIO_PORT8	bxxxxxxxx	bxxx1xxxxx xxxxxxxxxx xxxxxx	M2IOSF Port8
IIO_PORT9	bxxxxxxxx	bxx1xxxxxx xxxxxxxxxx xxxxxx	M2IOSF Port9
LLCPREF_LOCAL_F	bxxxxxxxx	bxxxxxxxxx xxxxxxxx1x xxxxxx	Local LLC prefetch requests (from LLC) Filter Any local LLC prefetch to the LLC
LOCAL_F	bxxxxxxxx	bxxxxxxxxx xxxx1xxxxx xxxxxx	Transactions homed locally Filter Transaction whose address resides in the local MC.

Table 2-89. Unit Masks for LLC_LOOKUP (Sheet 3 of 4)

Extension	umask [15:8]	xtra [57:32]	Description
OTHER_REQ_F	bxxxxxxxx	bxxxxxxxxx xxxxxxxxx xxxx1x	Write Request Filter Writeback transactions to the LLC. This includes all write transactions
PREF_OR_DMND_REMOTE_F	bxxxxxxxx	bxxxxxxxxx xxxxxx1xxx xxxxxx	Remote non-snoop request Filter Non-snoop transactions to the LLC from remote agent
REMOTE_F	bxxxxxxxx	bxxxxxxxxx xxx1xxxxxx xxxxxx	Transactions homed remotely Filter Transaction whose address resides in a remote MC
REMOTE_SNOOP_F	bxxxxxxxx	bxxxxxxxxx xxxxx1xxxx xxxxxx	Remote snoop request Filter Snoop transactions to the LLC from remote agent
RFO_F	bxxxxxxxx	bxxxxxxxxx xxxxxxxxx xx1xxx	RFO Request Filter Local or remote RFO transactions to the LLC. This includes RFO prefetch.
DATA_READ_MISS	b00000001	0x1FC1	Data Read Misses
I	bxxxxxxx1	bxxxxxxxxx xxxxxxxxx xxxxxx	I State Miss
MISS_ALL	b00000001	0x1FE0	All Misses
SF_S	bxxxxxx1x	bxxxxxxxxx xxxxxxxxx xxxxxx	SnoopFilter - S State SF Hit Shared State
SF_E	bxxxxx1xx	bxxxxxxxxx xxxxxxxxx xxxxxx	SnoopFilter - E State SF Hit Exclusive State
SF_H	bxxxx1xxx	bxxxxxxxxx xxxxxxxxx xxxxxx	SnoopFilter - H State SF Hit HitMe State
S	bxxx1xxxx	bxxxxxxxxx xxxxxxxxx xxxxxx	S State Hit Shared State
E	bxx1xxxxx	bxxxxxxxxx xxxxxxxxx xxxxxx	E State Hit Exclusive State
M	bx1xxxxxx	bxxxxxxxxx xxxxxxxxx xxxxxx	M State Hit Modified State
F	b1xxxxxxx	bxxxxxxxxx xxxxxxxxx xxxxxx	F State Hit Forward State
ALL_REMOTE	b11111111	0x17E0	All transactions from Remote Agents
DATA_READ_ALL	b11111111	0x1FC1	Data Reads
DATA_READ_LOCAL	b11111111	0x841	Demand Data Reads, Core and LLC prefetches
RFO	b11111111	0x1BC8	All RFOs - Demand and Prefetches
RFO_LOCAL	b11111111	0x9C8	Locally HOMed RFOs - Demand and Prefetches

Table 2-89. Unit Masks for LLC_LOOKUP (Sheet 4 of 4)

Extension	umask [15:8]	xtra [57:32]	Description
WRITE_LOCAL	b11111111	0x842	Writes Requests that install or change a line in the LLC. Examples: Write backs from Core L2sandUPI.PrefetchesintotheLLC.'
WRITE_REMOTE	b11111111	0x17C2	Remote Writes

LLC_VICTIMS

- **Title:**
- **Category:** CACHE Events
- **Event Code:** 0x37
- **Register Restrictions :** 0-3
- **Definition:** Counts the number of lines that were victimized on a fill. This can be filtered by the state that the line was in.

Table 2-90. Unit Masks for LLC_VICTIMS (Sheet 1 of 2)

Extension	umask [15:8]	xtra [57:32]	Description
LOCAL_ONLY	bxxxxxxxx	bxx1xxxxx	Local Only
REMOTE_ONLY	bxxxxxxxx	b1xxxxxxx	Remote Only
LOCAL_M	b00000001	b00100000	Local - Lines in M State
M_STATE	bxxxxxxx1	bxxxxxxxx	Lines in M state
REMOTE_M	b00000001	b10000000	Remote - Lines in M State
E_STATE	bxxxxxx1x	bxxxxxxxx	Lines in E state
LOCAL_E	b00000010	b00100000	Local - Lines in E State
REMOTE_E	b00000010	b10000000	Remote - Lines in E State
LOCAL_S	b00000100	b00100000	Local - Lines in S State
REMOTE_S	b00000100	b10000000	Remote - Lines in S State
S_STATE	bxxxxx1xx	bxxxxxxxx	Lines in S State
LOCAL_F	b00001000	b00100000	Local - Lines in F State
ALL	b00001111	b00000000	All Lines Victimized
LOCAL_ALL	b00001111	b00100000	Local - All Lines
REMOTE_ALL	b00001111	b10000000	Remote - All Lines

Table 2-90. Unit Masks for LLC_VICTIMS (Sheet 2 of 2)

Extension	umask [15:8]	xtra [57:32]	Description
IO	bxxx1xxxx	bxxxxxxxx	IO traffic
IA	bxx1xxxxx	bxxxxxxxx	IA traffic

MISC

- **Title:**
- **Category:** MISC Events
- **Event Code:** 0x39
- **Register Restrictions :** 0-3
- **Definition:** Miscellaneous events in the Cbo.

Table 2-91. Unit Masks for MISC

Extension	umask [15:8]	Description
RSPI_WAS_FSE	bxxxxxxx1	Silent Snoop Eviction Counts the number of times when a Snoop hit in FSE states and triggered a silent eviction. This is useful because this information is lost in the PRE encodings.
WC_ALIASING	bxxxxxx1x	Write Combining Aliasing Counts the number of times that a USWC write (WCIL(F)) transaction hit in the LLC in M state, triggering a WBMtoI followed by the USWC write. This occurs when there is WC aliasing.
RFO_HIT_S	bxxxx1xxx	RFO HitS Number of times that an RFO hit in S state. This is useful for determining if it might be good for a workload to use RspIWB instead of RspSWB.
CV0_PREF_VIC	bxxx1xxxx	CV0 Prefetch Victim
CV0_PREF_MISS	bxx1xxxxx	CV0 Prefetch Miss

MISC2

- **Title:**
- **Category:** MISC Events
- **Event Code:** 0x6A
- **Register Restrictions :** 0-3
- **Definition:** More Miscellaneous events in the Cbo.

Table 2-92. Unit Masks for MISC2 (Sheet 1 of 2)

Extension	umask [15:8]	Description
GOTRK_INSERT	bxxxxxxx1	GO Tracker Insert
GOTRCK_IN	bxxxxxx1x	GO Tracker Increment
RMW_LATE_SF_INSERT	bxxxxx1xx	Late Snoop Filter Allocation

Table 2-92. Unit Masks for MISC2 (Sheet 2 of 2)

Extension	umask [15:8]	Description
RMW_SF_LATE_CV	bxxxx1xxx	Late Snoop Filter CV Update
RMW_LLC_LATE_CV	bxxx1xxxx	Late LLC CV Update
GT_ONE_AKC_CRD	bxx1xxxxx	More than one AKC Credits

OSB

- **Title:**
- **Category:** HA OSB Events
- **Event Code:** 0x55
- **Register Restrictions :** 0-3
- **Definition:** Count of OSB snoop broadcasts. Counts by 1 per request causing OSB snoops to be broadcast. Does not count all the snoops generated by OSB.

Table 2-93. Unit Masks for OSB

Extension	umask [15:8]	Description
LOCAL_INVITOE	bxxxxxxx1	Local InvItoE
LOCAL_READ	bxxxxxx1x	Local Rd
REMOTE_READ	bxxxxx1xx	Remote Rd
REMOTE_READINVITOE	bxxxx1xxx	Remote Rd InvItoE
RFO_HITS_SNP_BCAST	bxxx1xxxx	RFO HitS Snoop Broadcast
OFF_PWRHEURISTIC	bxx1xxxxx	Off

PMM_MEMMODE_NM_INVITOX

- **Title:**
- **Category:** HA PM MEMMODE Events
- **Event Code:** 0x65
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-94. Unit Masks for PMM_MEMMODE_NM_INVITOX

Extension	umask [15:8]	Description
LOCAL	bxxxxxxx1	
REMOTE	bxxxxxx1x	
SETCONFLICT	bxxxxx1xx	

PMM_MEMMODE_NM_SETCONFLICTS

- **Title:**
- **Category:** HA PM MEMMODE Events
- **Event Code:** 0x64
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-95. Unit Masks for PMM_MEMMODE_NM_SETCONFLICTS

Extension	umask [15:8]	Description
SF	bxxxxxxx1	Counts the number of times CHA saw NM Set conflict in SF/LLC NM evictions due to another read to the same near memory set in the SF.
LLC	bxxxxxx1x	Counts the number of times CHA saw NM Set conflict in SF/LLC NM evictions due to another read to the same near memory set in the LLC.
TOR	bxxxxx1xx	Counts the number of times CHA saw NM Set conflict in TOR No Reject in the CHA due to a pending read to the same near memory set in the TOR.
TOR_REJECT	bxxxx1xxx	Counts the number of times CHA saw NM Set conflict in TOR and the transaction was rejected Rejects in the CHA due to a pending read to the same near memory set in the TOR.

PMM_QOS

- **Title:**
- **Category:** HA PMM QOS Events
- **Event Code:** 0x66
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-96. Unit Masks for PMM_QOS

Extension	umask [15:8]	Description
SLOW_INSERT	bxxxxxxx1	
DDR4_FAST_INSERT	bxxxxxx1x	
THROTTLE	bxxxxx1xx	
REJ_IRQ	bxxxx1xxx	
THROTTLE_PRQ	bxxx1xxxx	
THROTTLE_IRQ	bxx1xxxxx	
SLOWTORQ_SKIP	bx1xxxxxx	

PMM_QOS_OCCUPANCY

- **Title:**
- **Category:** HA PMM QOS Events
- **Event Code:** 0x67
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-97. Unit Masks for PMM_QOS_OCCUPANCY

Extension	umask [15:8]	Description
DDR_SLOW_FIFO	bxxxxxxx1	count the number of SLOW TOR Request inserted
DDR_FAST_FIFO	bxxxxxx1x	count the number of FAST TOR Request inserted

READ_NO_CREDITS

- **Title:**
- **Category:** MC Credit and Traffic Events
- **Event Code:** 0x58
- **Register Restrictions :** 0-3
- **Definition:** Counts the number of times when there are no credits available for sending reads from the CHA into the iMC. In order to send reads into the memory controller, the HA must first acquire a credit for the iMCs AD Ingress queue.

Table 2-98. Unit Masks for READ_NO_CREDITS

Extension	umask [15:8]	xtra [57:32]	Description
MC0	bxxxxxxx1	bxxxxxxxx	MC0 Filter for memory controller 0 only.
MC1	bxxxxxx1x	bxxxxxxxx	MC1 Filter for memory controller 1 only.
MC2	bxxxxx1xx	bxxxxxxxx	MC2 Filter for memory controller 2 only.
MC3	bxxxx1xxx	bxxxxxxxx	MC3 Filter for memory controller 3 only.
MC4	bxxx1xxxx	bxxxxxxxx	MC4 Filter for memory controller 4 only.
MC5	bxx1xxxxx	bxxxxxxxx	MC5 Filter for memory controller 5 only.

REQUESTS

- **Title:**
- **Category:** HA Request Events
- **Event Code:** 0x50
- **Register Restrictions :** 0-3
- **Definition:** Counts the total number of read requests made into the Home Agent. Reads include all read opcodes (including RFO). Writes include all writes (streaming, evictions, HitM, and so on).

Table 2-99. Unit Masks for REQUESTS

Extension	umask [15:8]	Description
READS_LOCAL	bxxxxxxx1	Reads Local Local read requests that miss the SF/LLC and are sent to the CHAs Home Agent'
READS_REMOTE	bxxxxxx1x	Reads Remote Remote read requests sent to the CHAs Home Agent'
READS	b00000011	Reads Local read requests that miss the SF/LLC and remote read requests sent to the CHAs Home Agent'
WRITES_LOCAL	bxxxxx1xx	Writes Local Local write requests that miss the SF/LLC and are sent to the CHAs HomeA gent'
WRITES_REMOTE	bxxxx1xxx	Writes Remote Remote write requests sent to the CHAs Home Agent'
WRITES	b00001100	Writes Local write requests that miss the SF/LLC and remote write requests sent to the CHAs Home Agent'
INVITOE_LOCAL	bxxx1xxxx	InvalItoE Local Local InvItoE requests (exclusive ownership of a cache line without receiving data) that miss the SF/LLC and are sent to the CHAs home agent'
INVITOE_REMOTE	bxx1xxxxx	InvalItoE Remote Remote InvItoE requests (exclusive ownership of a cache line without receiving data) sent to the CHAs home agent'
INVITOE	b00110000	InvalItoE InvItoE requests (exclusive ownership of a cache line without receiving data) sent to the CHAs home agent'

RxC_INSERTS

- **Title:**
- **Category:** Ingress Events
- **Event Code:** 0x13
- **Register Restrictions :** 0-3
- **Definition:** Counts number of allocations per cycle into the specified Ingress queue.

Table 2-100. Unit Masks for RxC_INSERTS (Sheet 1 of 2)

Extension	umask [15:8]	Description
IRQ	bxxxxxxx1	IRQ
IRQ_REJ	bxxxxxx1x	IRQ Rejected
IPQ	bxxxxx1xx	IPQ
PRQ	bxxx1xxxx	PRQ
PRQ_REJ	bxx1xxxxx	PRQ

Table 2-100. Unit Masks for RxC_INSERTS (Sheet 2 of 2)

Extension	umask [15:8]	Description
RRQ	bx1xxxxxx	RRQ
WBQ	b1xxxxxxx	WBQ

RxC_IPQ0_REJECT

- **Title:**
- **Category:** Ingress Reject and Retry Events
- **Event Code:** 0x22
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-101. Unit Masks for RxC_IPQ0_REJECT

Extension	umask [15:8]	Description
AD_REQ_VN0	bxxxxxxx1	AD REQ on VN0 No AD VN0 credit for generating a request
AD_RSP_VN0	bxxxxxx1x	AD RSP on VN0 No AD VN0 credit for generating a response
BL_RSP_VN0	bxxxxx1xx	BL RSP on VN0 No BL VN0 credit for generating a response
BL_WB_VN0	bxxxx1xxx	BL WB on VN0 No BL VN0 credit for generating a write back
BL_NCB_VN0	bxxx1xxxx	BL NCB on VN0 No BL VN0 credit for NCB
BL_NCS_VN0	bxx1xxxxx	BL NCS on VN0 No BL VN0 credit for NCS
AK_NON_UPI	bx1xxxxxx	Non UPI AK Request Cant inject AK ring message
IV_NON_UPI	b1xxxxxxx	Non UPI IV Request Cant inject IV ring message

RxC_IPQ1_REJECT

- **Title:**
- **Category:** Ingress Reject and Retry Events
- **Event Code:** 0x23
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-102. Unit Masks for RxC_IPQ1_REJECT (Sheet 1 of 2)

Extension	umask [15:8]	Description
ANY0	bxxxxxxx1	ANY0 Any condition listed in the IPQ0 Reject counter was true
HA	bxxxxxx1x	HA

Table 2-102. Unit Masks for RxC_IPQ1_REJECT (Sheet 2 of 2)

Extension	umask [15:8]	Description
LLC_VICTIM	bxxxxx1xx	LLC Victim
SF_VICTIM	bxxxx1xxx	SF Victim Requests did not generate Snoop filter victim
VICTIM	bxxx1xxxx	Victim
LLC_OR_SF_WAY	bxx1xxxxx	LLC OR SF Way Way conflict with another request that caused the reject
ALLOW_SNP	bx1xxxxxx	Allow Snoop
PA_MATCH	b1xxxxxxx	Phy Addr Match Address match with an outstanding request that was rejected.

RxC_IRQ0_REJECT

- **Title:**
- **Category:** Ingress Reject and Retry Events
- **Event Code:** 0x18
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-103. Unit Masks for RxC_IRQ0_REJECT

Extension	umask [15:8]	Description
AD_REQ_VN0	bxxxxxxx1	AD REQ on VN0 No AD VN0 credit for generating a request
AD_RSP_VN0	bxxxxxx1x	AD RSP on VN0 No AD VN0 credit for generating a response
BL_RSP_VN0	bxxxxx1xx	BL RSP on VN0 No BL VN0 credit for generating a response
BL_WB_VN0	bxxxx1xxx	BL WB on VN0 No BL VN0 credit for generating a write back
BL_NCB_VN0	bxxx1xxxx	BL NCB on VN0 No BL VN0 credit for NCB
BL_NCS_VN0	bxx1xxxxx	BL NCS on VN0 No BL VN0 credit for NCS
AK_NON_UPI	bx1xxxxxx	Non UPI AK Request Cant inject AK ring message
IV_NON_UPI	b1xxxxxxx	Non UPI IV Request Cant inject IV ring message

RxC_IRQ1_REJECT

- **Title:**
- **Category:** Ingress Reject and Retry Events
- **Event Code:** 0x19
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-104. Unit Masks for RxC_IRQ1_REJECT

Extension	umask [15:8]	Description
ANY0	bxxxxxxx1	ANY0 Any condition listed in the IRQ0 Reject counter was true
HA	bxxxxxx1x	HA
LLC_VICTIM	bxxxxx1xx	LLC Victim
SF_VICTIM	bxxxx1xxx	SF Victim Requests did not generate Snoop filter victim
VICTIM	bxxx1xxxx	Victim
LLC_OR_SF_WAY	bxx1xxxxx	LLC or SF Way Way conflict with another request that caused the reject
ALLOW_SNP	bx1xxxxxx	Allow Snoop
PA_MATCH	b1xxxxxxx	Phy Addr Match Address match with an outstanding request that was rejected.

RxC_ISMQ0_REJECT

- **Title:**
- **Category:** Ingress Reject and Retry Events
- **Event Code:** 0x24
- **Register Restrictions :** 0-3
- **Definition:** Number of times a transaction flowing through the ISMQ had to retry. Transaction pass through the ISMQ as responses for requests that already exist in the Cbo. Some examples include: when data is returned or when snoop responses come back from the cores.

Table 2-105. Unit Masks for RxC_ISMQ0_REJECT

Extension	umask [15:8]	Description
AD_REQ_VN0	bxxxxxxx1	AD REQ on VN0 No AD VN0 credit for generating a request
AD_RSP_VN0	bxxxxxx1x	AD RSP on VN0 No AD VN0 credit for generating a response
BL_RSP_VN0	bxxxxx1xx	BL RSP on VN0 No BL VN0 credit for generating a response
BL_WB_VN0	bxxxx1xxx	BL WB on VN0 No BL VN0 credit for generating a write back
BL_NCB_VN0	bxxx1xxxx	BL NCB on VN0 No BL VN0 credit for NCB
BL_NCS_VN0	bxx1xxxxx	BL NCS on VN0 No BL VN0 credit for NCS
AK_NON_UPI	bx1xxxxxx	Non UPI AK Request Cant inject AK ring message
IV_NON_UPI	b1xxxxxxx	Non UPI IV Request Cant inject IV ring message

RxC_ISMQ0_RETRY

- **Title:**
- **Category:** Ingress Reject and Retry Events
- **Event Code:** 0x2C
- **Register Restrictions :** 0-3
- **Definition:** Number of times a transaction flowing through the ISMQ had to retry. Transaction pass through the ISMQ as responses for requests that already exist in the Cbo. Some examples include: when data is returned or when snoop responses come back from the cores.

Table 2-106. Unit Masks for RxC_ISMQ0_RETRY

Extension	umask [15:8]	Description
AD_REQ_VN0	bxxxxxxx1	AD REQ on VN0 No AD VN0 credit for generating a request
AD_RSP_VN0	bxxxxxx1x	AD RSP on VN0 No AD VN0 credit for generating a response
BL_RSP_VN0	bxxxxx1xx	BL RSP on VN0 No BL VN0 credit for generating a response
BL_WB_VN0	bxxxx1xxx	BL WB on VN0 No BL VN0 credit for generating a write back
BL_NCB_VN0	bxxx1xxxx	BL NCB on VN0 No BL VN0 credit for NCB
BL_NCS_VN0	bxx1xxxxx	BL NCS on VN0 No BL VN0 credit for NCS
AK_NON_UPI	bx1xxxxxx	Non UPI AK Request Cant inject AK ring message
IV_NON_UPI	b1xxxxxxx	Non UPI IV Request Cant inject IV ring message

RxC_ISMQ1_REJECT

- **Title:**
- **Category:** Ingress Reject and Retry Events
- **Event Code:** 0x25
- **Register Restrictions :** 0-3
- **Definition:** Number of times a transaction flowing through the ISMQ had to retry. Transaction pass through the ISMQ as responses for requests that already exist in the Cbo. Some examples include: when data is returned or when snoop responses come back from the cores.

Table 2-107. Unit Masks for RxC_ISMQ1_REJECT

Extension	umask [15:8]	Description
ANY0	bxxxxxxx1	ANY0 Any condition listed in the ISMQ0 Reject counter was true
HA	bxxxxxx1x	HA

RxC_ISMQ1_RETRY

- **Title:**
- **Category:** Ingress Reject and Retry Events
- **Event Code:** 0x2D
- **Register Restrictions :** 0-3
- **Definition:** Number of times a transaction flowing through the ISMQ had to retry. Transaction pass through the ISMQ as responses for requests that already exist in the Cbo. Some examples include: when data is returned or when snoop responses come back from the cores.

Table 2-108. Unit Masks for RxC_ISMQ1_RETRY

Extension	umask [15:8]	Description
ANY0	bxxxxxxx1	ANY0 Any condition listed in the ISMQ0 Reject counter was true
HA	bxxxxxx1x	HA

RxC_OCCUPANCY

- **Title:**
- **Category:** Ingress Events
- **Event Code:** 0x11
- **Register Restrictions :** 0
- **Definition:** Counts number of entries in the specified Ingress queue in each cycle.

Table 2-109. Unit Masks for RxC_OCCUPANCY

Extension	umask [15:8]	Description
IPQ	b00000100	IPQ
RRQ	b01000000	RRQ
WBQ	b10000000	WBQ

RxC_OTHER0_RETRY

- **Title:**
- **Category:** Ingress Reject and Retry Events
- **Event Code:** 0x2E
- **Register Restrictions :** 0-3
- **Definition:** Retry Queue Inserts of Transactions that were already in another Retry Q (sub-events encode the reason for the next reject)

Table 2-110. Unit Masks for RxC_OTHER0_RETRY

Extension	umask [15:8]	Description
AD_REQ_VN0	bxxxxxxx1	AD REQ on VN0 No AD VN0 credit for generating a request
AD_RSP_VN0	bxxxxxx1x	AD RSP on VN0 No AD VN0 credit for generating a response
BL_RSP_VN0	bxxxxx1xx	BL RSP on VN0 No BL VN0 credit for generating a response
BL_WB_VN0	bxxxx1xxx	BL WB on VN0 No BL VN0 credit for generating a write back
BL_NCB_VN0	bxxx1xxxx	BL NCB on VN0 No BL VN0 credit for NCB
BL_NCS_VN0	bxx1xxxxx	BL NCS on VN0 No BL VN0 credit for NCS
AK_NON_UPI	bx1xxxxxx	Non UPI AK Request CantinjectAKringmessage'
IV_NON_UPI	b1xxxxxxx	Non UPI IV Request CantinjectIVringmessage'

RxC_OTHER1_RETRY

- **Title:**
- **Category:** Ingress Reject and Retry Events
- **Event Code:** 0x2F
- **Register Restrictions :** 0-3
- **Definition:** Retry Queue Inserts of Transactions that were already in another Retry Q (sub-events encode the reason for the next reject).

Table 2-111. Unit Masks for RxC_OTHER1_RETRY

Extension	umask [15:8]	Description
ANY0	bxxxxxxx1	ANY0 Any condition listed in the Other0 Reject counter was true
HA	bxxxxxx1x	HA
LLC_VICTIM	bxxxxx1xx	LLC Victim
SF_VICTIM	bxxxx1xxx	SF Victim Requests did not generate Snoop filter victim
VICTIM	bxxx1xxxx	Victim
LLC_OR_SF_WAY	bxx1xxxxx	LLC OR SF Way Way conflict with another request that caused the reject
ALLOW_SNP	bx1xxxxxx	Allow Snoop
PA_MATCH	b1xxxxxxx	PhyAddr Match Address match with an outstanding request that was rejected.

RxC_PRQ0_REJECT

- **Title:**
- **Category:** Ingress Reject and Retry Events
- **Event Code:** 0x20
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-112. Unit Masks for RxC_PRQ0_REJECT

Extension	umask [15:8]	Description
AD_REQ_VN0	bxxxxxxx1	AD REQ on VN0 No AD VN0 credit for generating a request
AD_RSP_VN0	bxxxxxx1x	AD RSP on VN0 No AD VN0 credit for generating a response
BL_RSP_VN0	bxxxxx1xx	BL RSP on VN0 No BL VN0 credit for generating a response
BL_WB_VN0	bxxxx1xxx	BL WB on VN0 No BL VN0 credit for generating a write back
BL_NCB_VN0	bxxx1xxxx	BL NCB on VN0 No BL VN0 credit for NCB
BL_NCS_VN0	bxx1xxxxx	BL NCS on VN0 No BL VN0 credit for NCS
AK_NON_UPI	bx1xxxxxx	Non UPI AK Request CantinjectAKringmessage'
IV_NON_UPI	b1xxxxxxx	Non UPI IV Request CantinjectIVringmessage'

RxC_PRQ1_REJECT

- **Title:**
- **Category:** Ingress Reject and Retry Events
- **Event Code:** 0x21
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-113. Unit Masks for RxC_PRQ1_REJECT (Sheet 1 of 2)

Extension	umask [15:8]	Description
ANY0	bxxxxxxx1	ANY0 Any condition listed in the PRQ0 Reject counter was true
HA	bxxxxxx1x	HA
LLC_VICTIM	bxxxxx1xx	LLC Victim
SF_VICTIM	bxxxx1xxx	SF Victim Requests did not generate Snoop filter victim
VICTIM	bxxx1xxxx	Victim
LLC_OR_SF_WAY	bxx1xxxxx	LLC OR SF Way Way conflict with another request that caused the reject

Table 2-113. Unit Masks for RxC_PRQ1_REJECT (Sheet 2 of 2)

Extension	umask [15:8]	Description
ALLOW_SNP	bx1xxxxxx	Allow Snoop
PA_MATCH	b1xxxxxxx	PhyAddr Match Address match with an outstanding request that was rejected.

RxC_REQ_Q0_RETRY

- **Title:**
- **Category:** Ingress Reject and Retry Events
- **Event Code:** 0x2A
- **Register Restrictions :** 0-3
- **Definition:** “REQUESTQ” includes: IRQ, PRQ, IPQ, RRQ, WBQ (everything except for ISMQ).

Table 2-114. Unit Masks for RxC_REQ_Q0_RETRY

Extension	umask [15:8]	Description
AD_REQ_VN0	bxxxxxxx1	AD REQ on VN0 No AD VN0 credit for generating a request
AD_RSP_VN0	bxxxxxx1x	AD RSP on VN0 No AD VN0 credit for generating a response
BL_RSP_VN0	bxxxxx1xx	BL RSP on VN0 No BL VN0 credit for generating a response
BL_WB_VN0	bxxxx1xxx	BL WB on VN0 No BL VN0 credit for generating a write back
BL_NCB_VN0	bxxx1xxxx	BL NCB on VN0 No BL VN0 credit for NCB
BL_NCS_VN0	bxx1xxxxx	BL NCS on VN0 No BL VN0 credit for NCS
AK_NON_UPI	bx1xxxxxx	Non UPI AK Request CantinjectAKringmessage'
IV_NON_UPI	b1xxxxxxx	Non UPI IV Request CantinjectIVringmessage'

RxC_REQ_Q1_RETRY

- **Title:**
- **Category:** Ingress Reject and Retry Events
- **Event Code:** 0x2B
- **Register Restrictions :** 0-3
- **Definition:** “REQUESTQ” includes: IRQ, PRQ, IPQ, RRQ, WBQ (everything except for ISMQ).

Table 2-115. Unit Masks for RxC_REQ_Q1_RETRY

Extension	umask [15:8]	Description
ANY0	bxxxxxxx1	ANY0 Any condition listed in the WBQ0 Reject counter was true
HA	bxxxxxx1x	HA
LLC_VICTIM	bxxxxx1xx	LLC Victim
SF_VICTIM	bxxxx1xxx	SF Victim Requests did not generate Snoop filter victim
VICTIM	bxxx1xxxx	Victim
LLC_OR_SF_WAY	bxx1xxxxx	LLC OR SF Way Way conflict with another request that caused the reject
ALLOW_SNP	bx1xxxxxx	Allow Snoop
PA_MATCH	b1xxxxxxx	PhyAddr Match Address match with an outstanding request that was rejected.

RxC_RRQ0_REJECT

- **Title:**
- **Category:** Ingress Reject and Retry Events
- **Event Code:** 0x26
- **Register Restrictions :** 0-3
- **Definition:** Number of times a transaction flowing through the Remote Response Queue (RRQ) had to retry.

Table 2-116. Unit Masks for RxC_RRQ0_REJECT

Extension	umask [15:8]	Description
AD_REQ_VN0	bxxxxxxx1	AD REQ on VN0 No AD VN0 credit for generating a request
AD_RSP_VN0	bxxxxxx1x	AD RSP on VN0 No AD VN0 credit for generating a response
BL_RSP_VN0	bxxxxx1xx	BL RSP on VN0 No BL VN0 credit for generating a response
BL_WB_VN0	bxxxx1xxx	BL WB on VN0 No BL VN0 credit for generating a write back
BL_NCB_VN0	bxxx1xxxx	BL NCB on VN0 No BL VN0 credit for NCB
BL_NCS_VN0	bxx1xxxxx	BL NCS on VN0 No BL VN0 credit for NCS
AK_NON_UPI	bx1xxxxxx	Non UPI AK Request CantinjectAKringmessage'
IV_NON_UPI	b1xxxxxxx	Non UPI IV Request CantinjectIVringmessage'

RxC_RRQ1_REJECT

- **Title:**
- **Category:** Ingress Reject and Retry Events
- **Event Code:** 0x27
- **Register Restrictions :** 0-3
- **Definition:** Number of times a transaction flowing through the Remote Response Queue (RRQ) had to retry.

Table 2-117. Unit Masks for RxC_RRQ1_REJECT

Extension	umask [15:8]	Description
ANY0	bxxxxxxx1	ANY0 Any condition listed in the RRQ0 Reject counter was true
HA	bxxxxxx1x	HA
LLC_VICTIM	bxxxxx1xx	LLC Victim
SF_VICTIM	bxxxx1xxx	SF Victim Requests did not generate Snoop filter victim
VICTIM	bxxx1xxxx	Victim
LLC_OR_SF_WAY	bxx1xxxxx	LLC OR SF Way Way conflict with another request that caused the reject
ALLOW_SNP	bx1xxxxxx	Allow Snoop
PA_MATCH	b1xxxxxxx	PhyAddr Match Address match with an outstanding request that was rejected.

RxC_WBQ0_REJECT

- **Title:**
- **Category:** Ingress Reject and Retry Events
- **Event Code:** 0x28
- **Register Restrictions :** 0-3
- **Definition:** Number of times a transaction flowing through the Write Back Queue (WBQ) had to retry.

Table 2-118. Unit Masks for RxC_WBQ0_REJECT (Sheet 1 of 2)

Extension	umask [15:8]	Description
AD_REQ_VN0	bxxxxxxx1	AD REQ on VN0 No AD VN0 credit for generating a request
AD_RSP_VN0	bxxxxxx1x	AD RSP on VN0 No AD VN0 credit for generating a response
BL_RSP_VN0	bxxxxx1xx	BL RSP on VN0 No BL VN0 credit for generating a response
BL_WB_VN0	bxxxx1xxx	BL WB on VN0 No BL VN0 credit for generating a write back
BL_NCB_VN0	bxxx1xxxx	BL NCB on VN0 No BL VN0 credit for NCB
BL_NCS_VN0	bxx1xxxxx	BL NCS on VN0 No BL VN0 credit for NCS

Table 2-118. Unit Masks for RxC_WBQ0_REJECT (Sheet 2 of 2)

Extension	umask [15:8]	Description
AK_NON_UPI	bx1xxxxxx	Non UPI AK Request CantinjectAKringmessage'
IV_NON_UPI	b1xxxxxxx	Non UPI IV Request CantinjectIVringmessage'

RxC_WBQ1_REJECT

- **Title:**
- **Category:** Ingress Reject and Retry Events
- **Event Code:** 0x29
- **Register Restrictions :** 0-3
- **Definition:** Number of times a transaction flowing through the Write Back Queue (WBQ) had to retry.

Table 2-119. Unit Masks for RxC_WBQ1_REJECT

Extension	umask [15:8]	Description
ANY0	bxxxxxxx1	ANY0 Any condition listed in the WBQ0 Reject counter was true
HA	bxxxxxx1x	HA
LLC_VICTIM	bxxxxx1xx	LLC Victim
SF_VICTIM	bxxxx1xxx	SF Victim Requests did not generate Snoop filter victim
VICTIM	bxxx1xxxx	Victim
LLC_OR_SF_WAY	bxx1xxxxx	LLC OR SF Way Way conflict with another request that caused the reject
ALLOW_SNP	bx1xxxxxx	Allow Snoop
PA_MATCH	b1xxxxxxx	PhyAddr Match Address match with an outstanding request that was rejected.

SF_EVICTION

- **Title:**
- **Category:** CACHE Events
- **Event Code:** 0x3D
- **Register Restrictions :**
- **Definition:** Counts number of times a snoop filter entry was evicted, due to lack of space, and replaced with a new entry.

Table 2-120. Unit Masks for SF_EVICTION

Extension	umask [15:8]	Description
M_STATE	bxxxxxxx1	M state
E_STATE	bxxxxxx1x	E state
S_STATE	bxxxxx1xx	S state

SNOOPS_SENT

- **Title:**
- **Category:** HA Request Events
- **Event Code:** 0x51
- **Register Restrictions :** 0-3
- **Definition:** Counts the number of snoops issued by the HA.

Table 2-121. Unit Masks for SNOOPS_SENT

Extension	umask [15:8]	Description
ALL	bxxxxxxx1	All
LOCAL	bxxxxx1xx	Snoops sent for Local Requests Counts the number of broadcast or directed snoops issued by the HA responding to local requests
REMOTE	bxxxx1xxx	Snoops sent for Remote Requests Counts the number of broadcast or directed snoops issued by the HA responding to remote requests
BCST_LOCAL	bxxx1xxxx	Broadcast snoops for Local Requests Counts the number of broadcast snoops issued by the HA responding to local requests
BCST_REMOTE	bxx1xxxxx	Broadcast snoops for Remote Requests Counts the number of broadcast snoops issued by the HA responding to remote requests
DIRECT_LOCAL	bx1xxxxxx	Directed snoops for Local Requests Counts the number of directed snoops issued by the HA responding to local requests
DIRECT_REMOTE	b1xxxxxxx	Directed snoops for Remote Requests Counts the number of directed snoops issued by the HA responding to remote requests

SNOOP_RESP

- **Title:**
- **Category:** HA Snoop Response Events
- **Event Code:** 0x5C
- **Register Restrictions :** 0-3
- **Definition:** Counts the total number of RspI snoop responses received. Whenever a snoops are issued, one or more snoop responses will be returned depending on the topology of the system. In systems larger than 2s, when multiple snoops are returned this will count all the snoops that are received. For example, if three snoops were issued and returned RspI, RspS, and RspSFwd; then each of these sub-events would increment by 1.

Table 2-122. Unit Masks for SNOOP_RESP

Extension	umask [15:8]	Description
RSPI	bxxxxxxx1	RspI Filters for snoops responses of RspI. RspI is returned when the remote cache does not have the data, or when the remote cache silently evicts data (such as when an RFO hits non-modified data).
RSPS	bxxxxxx1x	RspS Filters for snoop responses of RspS. RspS is returned when a remote cache has data but is not forwarding it. It is a way to let the requesting socket know that it cannot allocate the data in E state. No data is sent with S RspS.
RSPIFWD	bxxxxx1xx	RspIFwd Filters for snoop responses of RspIFwd. This is returned when a remote caching agent forwards data and the requesting agent is able to acquire the data in E or M states. This is commonly returned with RFO transactions. It can be either a HitM or a HitFE.
RSPSFWD	bxxxx1xxx	RspSFwd Filters for a snoop response of RspSFwd. This is returned when a remote caching agent forwards data but holds on to its current copy. This is common for data and code reads that hit in a remote socket in E or F state.
RSPWB	bxxx1xxxx	Rsp*WB Filters for a snoop response of RspIWB or RspSWB. This is returned when a non-RFO request hits in M state. Data and Code Reads can return either RspIWB or RspSWB depending on how the system has been configured. InvItoE transactions will also return RspIWB because they must acquire ownership.
RSPFWDWB	bxx1xxxxx	Rsp*Fwd*WB Filters for a snoop response of Rsp*Fwd*WB. This snoop response is only used in 4s systems. It is used when a snoop HITMs in a remote caching agent and it directly forwards data to a requester, and simultaneously returns data to the home to be written back to memory.'
RSPCNFLCT	bx1xxxxxx	RSPCNFLCT* Filters for snoops responses of RspConflict. This is returned when a snoop finds an existing outstanding transaction in a remote caching agent when it CAMs that caching agent. This triggers conflict resolution hardware. This covers both RspCnflct and RspCnflctWbI.
RSPFWD	b1xxxxxxx	RspFwd Filters for a snoop response of RspFwd to a CA request. This snoop response is only possible for RdCur when a snoop HitMe in a remote caching agent and it directly forwards data to a requester without changing the requester's cacheline state.'

SNOOP_RESP_LOCAL

- **Title:**
- **Category:** HA Snoop Response Events
- **Event Code:** 0x5D
- **Register Restrictions :** 0-3
- **Definition:** Number of snoop responses received for a local request.

Table 2-123. Unit Masks for SNOOP_RESP_LOCAL

Extension	umask [15:8]	Description
RSPI	bxxxxxxx1	RspI Filters for snoops responses of RspI to local CA requests. RspI is returned when the remote cache does not have the data, or when the remote cache silently evicts data (such as when an RFO hits non-modified data).
RSPS	bxxxxxx1x	RspS Filters for snoop responses of RspS to local CA requests. RspS is returned when a remote cache has data but is not forwarding it. It is a way to let the requesting socket know that it cannot allocate the data in E state. No data is sent with S RspS.
RSPIFWD	bxxxxx1xx	RspIFwd Filters for snoop responses of RspIFwd to local CA requests. This is returned when a remote caching agent forwards data and the requesting agent is able to acquire the data in E or M states. This is commonly returned with RFO transactions. It can be either a HitM or a HitFE.
RSPSFWD	bxxxx1xxx	RspSFwd Filters for a snoop response of RspSFwd to local CA requests. This is returned when a remote caching agent forwards data but holds on to its currentl copy. This is common for data and code reads that hit in a remote socket in E or F state.
RSPWB	bxxx1xxxx	Rsp*WB Filters for a snoop response of RspIWB or RspSWB to local CA requests. This is returned when a non-RFO request hits in M state. Data and Code Reads can return either RspIWB or RspSWB depending on how the system has been configured. InvItoE transactions will also return RspIWB because they must acquire ownership.
RSPFWDWB	bxx1xxxxx	Rsp*FWD*WB Filters for a snoop response of Rsp*Fwd*WB to local CA requests. This snoop response is only used in 4s systems. It is used when a snoop HITMsinaremotecachingagentanditdirectlyforwardsdatatoarequestor,andsimultaneouslyreturnsdatatothehometobewrittenbacktomemory.'
RSPCNFLCT	bx1xxxxxx	RspCnflct Filters for snoops responses of RspConflict to local CA requests. This is returned when a snoop finds an existing outstanding transaction in a remote caching agent when it CAMs that caching agent. This triggers conflict resolution hardware. This covers both RspCnflct and RspCnflctWbI.
RSPFWD	b1xxxxxxx	RspFwd Filters for a snoop response of RspFwd to local CA requests. This snoop response is only possible for RdCur when a snoop HITM/E in a remote caching agent and it directly forwards data to a requestor without changing the requestorscachelinestate.'

TOR_INSERTS

- **Title:**
- **Category:** TOR Events
- **Event Code:** 0x35
- **Register Restrictions :** 0-3
- **Definition:** Counts the number of entries successfuly inserted into the TOR that match qualifications specified by the subevent. Does not include addressless requests such as locks and interrupts.

Table 2-124. Unit Masks for TOR_INSERTS (Sheet 1 of 6)

Extension	umask [15:8]	xtra [57:32]	Description
CXL_MEM_ACC	bxxxxxxxx	bxx1xxxxxxxxxxxxxxxxxxxxxxxxxx	
DDR	bxxxxxxxx	bxxxxxxxxxxxxxxxxxxxxxxxxx1xx	DDR4 Access
HBM	bxxxxxxxx	bxxxxxxxxxxxxxxxxxxxxxxx1xxxx	HBM Access
HIT	bxxxxxxxx	bxxxxxxxxxxxxxxxxxxxxxxxxxxx1	Just Hits
ISOC	bxxxxxxxx	bxxxxx1xxxxxxxxxxxxxxxxxxxxxx	Just ISOC
LOCAL_TGT	bxxxxxxxx	bxxxxxxxxxxxxxxxxxxxxxx1xxxxxxx	Just Local Targets
MATCH_OPC	bxxxxxxxx	bxxxxxxxxxxxxxxxxxxxx1xxxxxxxxx	Match the Opcode in b[29:19] of the extended umask field
MISS	bxxxxxxxx	bxxxxxxxxxxxxxxxxxxxxxxxxxxx1x	Just Misses
MMCFG	bxxxxxxxx	bxxxxxxxxxxxxxxxxxxxxxxxx1xxxxx	MMCFG Access
MMIO	bxxxxxxxx	bxxxxxxxxxxxxxxxxxxxxxxx1xxxxxx	MMIO Access
NEARMEM	bxxxxxxxx	bxxxxxxxxx1xxxxxxxxxxxxxxxxxxxx	Just NearMem
NONCOH	bxxxxxxxx	bxxxxxx1xxxxxxxxxxxxxxxxxxxxxx	Just NonCoherent
NOT_NEARMEM	bxxxxxxxx	bxxxxxxx1xxxxxxxxxxxxxxxxxxxxx	Just NotNearMem
PMM	bxxxxxxxx	bxxxxxxxxxxxxxxxxxxxxxxxxxx1xxx	PMM Access
PREMORPH_OPC	bxxxxxxxx	bxxxxxxxxxxxxxxxxxxx1xxxxxxxxxx	Match the PreMorphed Opcode in b[29:19] of the extended umask field
REMOTE_TGT	bxxxxxxxx	bxxxxxxxxxxxxxxxxxxxxx1xxxxxxxx	Just Remote Targets
IA	b00000001	0xC001FF	All requests from iA Cores
IA_CLFLUSH	b00000001	0xC8C7FF	CLFlushes issued by iA Cores
IA_CLFLUSHOPT	b00000001	0xC8D7FF	CLFlushOpts issued by iA Cores

Table 2-124. Unit Masks for TOR_INSERTS (Sheet 2 of 6)

Extension	umask [15:8]	xtra [57:32]	Description
IA_CRD	b00000001	0xC80FFF	CRDs issued by iA Cores
IA_DRD	b00000001	0xC817FF	DRds issued by iA Cores
IA_DRDPTE	b00000001	0xC837FF	DRdPte issued by iA Cores due to a page walk
IA_DRD_OPT	b00000001	0xC827FF	DRd_Opts issued by iA Cores
IA_DRD_OPT_PREF	b00000001	0xC8A7FF	DRd_Opt_Prefs issued by iA Cores
IA_DRD_PREF	b00000001	0xC897FF	DRd_Prefs issued by iA Cores
IA_HIT	b00000001	0xC001FD	All requests from iA Cores that Hit the LLC
IA_HIT_CRD	b00000001	0xC80FFD	CRds issued by iA Cores that Hit the LLC
IA_HIT_CRD_PREF	b00000001	0xC88FFD	CRd_Prefs issued by iA Cores that hit the LLC
IA_HIT_DRD	b00000001	0xC817FD	DRds issued by iA Cores that Hit the LLC
IA_HIT_DRDPTE	b00000001	0xC837FD	DRdPte issued by iA Cores due to a page walk that hit the LLC
IA_HIT_DRD_OPT	b00000001	0xC827FD	DRd_Opts issued by iA Cores that hit the LLC
IA_HIT_DRD_OPT_PREF	b00000001	0xC8A7FD	DRd_Opt_Prefs issued by iA Cores that hit the LLC
IA_HIT_DRD_PREF	b00000001	0xC897FD	DRd_Prefs issued by iA Cores that Hit the LLC
IA_HIT_ITOM	b00000001	0xCC47FD	ItoMs issued by iA Cores that Hit LLC
IA_HIT_LLCPREFCODE	b00000001	0xCCCFFD	LLCPrefCode issued by iA Cores that hit the LLC
IA_HIT_LLCPREFDATA	b00000001	0xCCD7FD	LLCPrefData issued by iA Cores that hit the LLC
IA_HIT_LLCPREFRFO	b00000001	0xCCC7FD	LLCPrefRFO issued by iA Cores that hit the LLC
IA_HIT_RFO	b00000001	0xC807FD	RFOs issued by iA Cores that Hit the LLC
IA_HIT_RFO_PREF	b00000001	0xC887FD	RFO_Prefs issued by iA Cores that Hit the LLC
IA_ITOM	b00000001	0xCC47FF	ItoMs issued by iA Cores
IA_ITOMCACHENEAR	b00000001	0xCD47FF	ItoMCacheNears issued by iA Cores
IA_LLCPREFCODE	b00000001	0xCCCFFF	LLCPrefCode issued by iA Cores
IA_LLCPREFDATA	b00000001	0xCCD7FF	LLCPrefData issued by iA Cores
IA_LLCPREFRFO	b00000001	0xCCC7FF	LLCPrefRFO issued by iA Cores

Table 2-124. Unit Masks for TOR_INSERTS (Sheet 3 of 6)

Extension	umask [15:8]	xtra [57:32]	Description
IA_MISS	b00000001	0xC001FE	All requests from iA Cores that Missed the LLC
IA_MISS_CRD	b00000001	0xC80FFE	CRds issued by iA Cores that Missed the LLC
IA_MISS_CRD_LOCAL	b00000001	0xC80EFE	CRd issued by iA Cores that Missed the LLC - HOMed locally
IA_MISS_CRD_PREF	b00000001	0xC88FFE	CRd_Prefs issued by iA Cores that Missed the LLC
IA_MISS_CRD_PREF_LOCAL	b00000001	0xC88EFE	CRd_Prefs issued by iA Cores that Missed the LLC - HOMed locally
IA_MISS_CRD_PREF_REMOTE	b00000001	0xC88F7E	CRd_Prefs issued by iA Cores that Missed the LLC - HOMed remotely
IA_MISS_CRD_REMOTE	b00000001	0xC80F7E	CRd issued by iA Cores that Missed the LLC - HOMed remotely
IA_MISS_DRD	b00000001	0xC817FE	DRds issued by iA Cores that Missed the LLC
IA_MISS_DRDPTE	b00000001	0xC837FE	DRdPte issued by iA Cores due to a page walk that missed the LLC
IA_MISS_DRD_DDR	b00000001	0xC81786	DRds issued by iA Cores targeting DDR Mem that Missed the LLC
IA_MISS_DRD_LOCAL	b00000001	0xC816FE	DRds issued by iA Cores that Missed the LLC - HOMed locally
IA_MISS_DRD_LOCAL_DDR	b00000001	0xC81686	DRds issued by iA Cores targeting DDR Mem that Missed the LLC - HOMed locally
IA_MISS_DRD_LOCAL_PMM	b00000001	0xC8168A	DRds issued by iA Cores targeting PMM Mem that Missed the LLC - HOMed locally
IA_MISS_DRD_OPT	b00000001	0xC827FE	DRd_Opt issued by iA Cores that missed the LLC
IA_MISS_DRD_OPT_PREF	b00000001	0xC8A7FE	DRd_Opt_Prefs issued by iA Cores that missed the LLC
IA_MISS_DRD_PMM	b00000001	0xC8178A	DRds issued by iA Cores targeting PMM Mem that Missed the LLC
IA_MISS_DRD_PREF	b00000001	0xC897FE	DRd_Prefs issued by iA Cores that Missed the LLC
IA_MISS_DRD_PREF_DDR	b00000001	0xC89786	DRd_Prefs issued by iA Cores targeting DDR Mem that Missed the LLC
IA_MISS_DRD_PREF_LOCAL_DDR	b00000001	0xC89686	DRd_Prefs issued by iA Cores targeting DDR Mem that Missed the LLC - HOMed locally
IA_MISS_DRD_PREF_LOCAL_PMM	b00000001	0xC8968A	DRd_Prefs issued by iA Cores targeting PMM Mem that Missed the LLC - HOMed locally
IA_MISS_DRD_PREF_PMM	b00000001	0xC8978A	DRd_Prefs issued by iA Cores targeting PMM Mem that Missed the LLC

Table 2-124. Unit Masks for TOR_INSERTS (Sheet 4 of 6)

Extension	umask [15:8]	xtra [57:32]	Description
IA_MISS_DRD_PREF_REMOTE_DDR	b00000001	0xC89706	DRd_Prefs issued by iA Cores targeting DDR Mem that Missed the LLC - HOMed remotely
IA_MISS_DRD_PREF_REMOTE_PMM	b00000001	0xC8970A	DRd_Prefs issued by iA Cores targeting PMM Mem that Missed the LLC - HOMed remotely
IA_MISS_DRD_REMOTE	b00000001	0xC8177E	DRds issued by iA Cores that Missed the LLC - HOMed remotely
IA_MISS_DRD_REMOTE_DDR	b00000001	0xC81706	DRds issued by iA Cores targeting DDR Mem that Missed the LLC - HOMed remotely
IA_MISS_DRD_REMOTE_PMM	b00000001	0xC8170A	DRds issued by iA Cores targeting PMM Mem that Missed the LLC - HOMed remotely
IA_MISS_ITOM	b00000001	0xCC47FE	ItoMs issued by iA Cores that Missed LLC
IA_MISS_LLCPREFCODE	b00000001	0xCCCFFE	LLCPrefCode issued by iA Cores that missed the LLC
IA_MISS_LLCPREFDATA	b00000001	0xCCD7FE	LLCPrefData issued by iA Cores that missed the LLC
IA_MISS_LLCPREFRFO	b00000001	0xCCC7FE	LLCPrefRFO issued by iA Cores that missed the LLC
IA_MISS_LOCAL_WCILF_DDR	b00000001	0xC86686	WCiLFs issued by iA Cores targeting DDR that missed the LLC - HOMed locally
IA_MISS_LOCAL_WCILF_PMM	b00000001	0xC8668A	WCiLFs issued by iA Cores targeting PMM that missed the LLC - HOMed locally
IA_MISS_LOCAL_WCIL_DDR	b00000001	0xC86E86	WCiLs issued by iA Cores targeting DDR that missed the LLC - HOMed locally
IA_MISS_LOCAL_WCIL_PMM	b00000001	0xC86E8A	WCiLs issued by iA Cores targeting PMM that missed the LLC - HOMed locally
IA_MISS_REMOTE_WCILF_DDR	b00000001	0xC86706	WCiLFs issued by iA Cores targeting DDR that missed the LLC - HOMed remotely
IA_MISS_REMOTE_WCILF_PMM	b00000001	0xC8670A	WCiLFs issued by iA Cores targeting PMM that missed the LLC - HOMed remotely
IA_MISS_REMOTE_WCIL_DDR	b00000001	0xC86F06	WCiLs issued by iA Cores targeting DDR that missed the LLC - HOMed remotely
IA_MISS_REMOTE_WCIL_PMM	b00000001	0xC86F0A	WCiLs issued by iA Cores targeting PMM that missed the LLC - HOMed remotely
IA_MISS_RFO	b00000001	0xC807FE	RFOs issued by iA Cores that Missed the LLC
IA_MISS_RFO_LOCAL	b00000001	0xC806FE	RFOs issued by iA Cores that Missed the LLC - HOMed locally
IA_MISS_RFO_PREF	b00000001	0xC887FE	RFO_Prefs issued by iA Cores that Missed the LLC

Table 2-124. Unit Masks for TOR_INSERTS (Sheet 5 of 6)

Extension	umask [15:8]	xtra [57:32]	Description
IA_MISS_RFO_PREF_LOCAL	b00000001	0xC886FE	RFO_Prefs issued by iA Cores that Missed the LLC - HOMed locally
IA_MISS_RFO_PREF_REMOTE	b00000001	0xC8877E	RFO_Prefs issued by iA Cores that Missed the LLC - HOMed remotely
IA_MISS_RFO_REMOTE	b00000001	0xC8077E	RFOs issued by iA Cores that Missed the LLC - HOMed remotely
IA_MISS_UCRDF	b00000001	0xC877DE	UCRdFs issued by iA Cores that Missed LLC
IA_MISS_WCIL	b00000001	0xC86FFE	WCiLs issued by iA Cores that Missed the LLC
IA_MISS_WCILF	b00000001	0xC867FE	WCiLF issued by iA Cores that Missed the LLC
IA_MISS_WCILF_DDR	b00000001	0xC86786	WCiLFs issued by iA Cores targeting DDR that missed the LLC
IA_MISS_WCILF_PMM	b00000001	0xC8678A	WCiLFs issued by iA Cores targeting PMM that missed the LLC
IA_MISS_WCIL_DDR	b00000001	0xC86F86	WCiLs issued by iA Cores targeting DDR that missed the LLC
IA_MISS_WCIL_PMM	b00000001	0xC86F8A	WCiLs issued by iA Cores targeting PMM that missed the LLC
IA_MISS_WIL	b00000001	0xC87FDE	WiLs issued by iA Cores that Missed LLC
IA_RFO	b00000001	0xC807FF	RFOs issued by iA Cores
IA_RFO_PREF	b00000001	0xC887FF	RFO_Prefs issued by iA Cores
IA_SPECITOM	b00000001	0xCC57FF	SpecItoMs issued by iA Cores
IA_WBMTOI	b00000001	0xCC27FF	WbMtoIs issued by iA Cores
IA_WCIL	b00000001	0xC86FFF	WCiLs issued by iA Cores
IA_WCILF	b00000001	0xC867FF	WCiLF issued by iA Cores
IRQ_IA	bxxxxxxx1	bxxxxxxxxxx xxxxxxxxxxx xxxxxxxxxx	IRQ - iA From an iA Core
LOC_IA	b00000001	0xC000FF	All from Local iA All locally initiated requests from iA Cores
EVICT	bxxxxxx1x	bxxxxxxxxxx xxxxxxxxxxx xxxxxxxxxx	SF/LLC Evictions TOR allocation occurred as a result of SF/LLC evictions (came from the ISMQ)
IO	b00000100	0xC001FF	All requests from IO Devices
IO_CLFLUSH	b00000100	0xC8C3FF	CLFlushes issued by IO Devices
IO_HIT	b00000100	0xC001FD	All requests from IO Devices that hit the LLC
IO_HIT_ITOM	b00000100	0xCC43FD	ItoMs issued by IO Devices that Hit the LLC

Table 2-124. Unit Masks for TOR_INSERTS (Sheet 6 of 6)

Extension	umask [15:8]	xtra [57:32]	Description
IO_HIT_ITOMCACHENEAR	b00000100	0xCD43FD	ItoMCacheNears, indicating a partial write request, from IO Devices that hit the LLC
IO_HIT_PCIRDCUR	b00000100	0xC8F3FD	PCIRdCurs issued by IO Devices that hit the LLC
IO_HIT_RFO	b00000100	0xC803FD	RFOs issued by IO Devices that hit the LLC
IO_ITOM	b00000100	0xCC43FF	ItoMs issued by IO Devices
IO_ITOMCACHENEAR	b00000100	0xCD43FF	ItoMCacheNears, indicating a partial write request, from IO Devices
IO_MISS	b00000100	0xC001FE	All requests from IO Devices that missed the LLC
IO_MISS_ITOM	b00000100	0xCC43FE	ItoMs issued by IO Devices that missed the LLC
IO_MISS_ITOMCACHENEAR	b00000100	0xCD43FE	ItoMCacheNears, indicating a partial write request, from IO Devices that missed the LLC
IO_MISS_PCIRDCUR	b00000100	0xC8F3FE	PCIRdCurs issued by IO Devices that missed the LLC
IO_MISS_RFO	b00000100	0xC803FE	RFOs issued by IO Devices that missed the LLC
IO_PCIRDCUR	b00000100	0xC8F3FF	PCIRdCurs issued by IO Devices
IO_RFO	b00000100	0xC803FF	RFOs issued by IO Devices
IO_WBMTOI	b00000100	0xCC23FF	WbMtoIs issued by IO Devices
LOC_IO	b00000100	0xC000FF	All from Local IO All locally generated IO traffic
PRQ_IOSF	bxxxxx1xx	bxxxxxxxxxx xxxxxxxxxx xxxxxxxxxx	PRQ - IOSF From a PCIe Device
LOC_ALL	b00000101	0xC000FF	All from Local iA and IO All locally initiated requests
IPQ	bxxxx1xxx	bxxxxxxxxxx xxxxxxxxxx xxxxxxxxxx	IPQ
IRQ_NON_IA	bxxx1xxxx	bxxxxxxxxxx xxxxxxxxxx xxxxxxxxxx	IRQ - Non iA
PRQ_NON_IOSF	bxx1xxxxx	bxxxxxxxxxx xxxxxxxxxx xxxxxxxxxx	PRQ - Non IOSF
RRQ	bx1xxxxxx	bxxxxxxxxxx xxxxxxxxxx xxxxxxxxxx	RRQ
WBQ	b1xxxxxxx	bxxxxxxxxxx xxxxxxxxxx xxxxxxxxxx	WBQ
ALL	b11111111	0xC001FF	All

TOR_OCCUPANCY

- **Title:**
- **Category:** TOR Events
- **Event Code:** 0x36
- **Register Restrictions :** 0
- **Definition:** For each cycle, this event accumulates the number of valid entries in the TOR that match qualifications specified by the subevent. Does not include addressless requests such as locks and interrupts.

Table 2-125. Unit Masks for TOR_OCCUPANCY (Sheet 1 of 6)

Extension	umask [15:8]	xtra [57:32]	Description
CXL_MEM_ACC	bxxxxxxxx	bxx1xxxxxxx xxxxxxxxxxx xxxxxxxxxx	
DDR	bxxxxxxxx	bxxxxxxxxxx xxxxxxxxxxx xxxxxxx1xx	DDR4 Access
HBM	bxxxxxxxx	bxxxxxxxxxx xxxxxxxxxxx xxxxx1xxxx	HBM Access
HIT	bxxxxxxxx	bxxxxxxxxxx xxxxxxxxxxx xxxxxxxxx1	Just Hits
ISOC	bxxxxxxxx	bxxxxx1xxxx xxxxxxxxxxx xxxxxxxxxx	Just ISOC
LOCAL_TGT	bxxxxxxxx	bxxxxxxxxxx xxxxxxxxxxx xx1xxxxxxx	Just Local Targets
MATCH_OPC	bxxxxxxxx	bxxxxxxxxxx xxxxxxxxxxx 1xxxxxxxxx	Match the Opcode in b[29:19] of the extended umask field
MISS	bxxxxxxxx	bxxxxxxxxxx xxxxxxxxxxx xxxxxxxx1x	Just Misses
MMCFG	bxxxxxxxx	bxxxxxxxxxx xxxxxxxxxxx xxxx1xxxxx	MMCFG Access
MMIO	bxxxxxxxx	bxxxxxxxxxx xxxxxxxxxxx xxx1xxxxxx	MMIO Access
NEARMEM	bxxxxxxxx	bxxxxxxxx1x xxxxxxxxxxx xxxxxxxxxx	Just NearMem
NONCOH	bxxxxxxxx	bxxxxxx1xxx xxxxxxxxxxx xxxxxxxxxx	Just NonCoherent
NOT_NEARMEM	bxxxxxxxx	bxxxxxxx1xx xxxxxxxxxxx xxxxxxxxxx	Just NotNearMem
PMM	bxxxxxxxx	bxxxxxxxxxx xxxxxxxxxxx xxxxxx1xxx	PMM Access
PREMORPH_OPC	bxxxxxxxx	bxxxxxxxxxx xxxxxxxxxx1 xxxxxxxxxx	Match the PreMorphed Opcode in b[29:19] of the extended umask field

Table 2-125. Unit Masks for TOR_OCCUPANCY (Sheet 2 of 6)

Extension	umask [15:8]	xtra [57:32]	Description
REMOTE_TGT	bxxxxxxxx	bxxxxxxxxxx xxxxxxxxxxx x1xxxxxxxx	Just Remote Targets
IA	b00000001	0xC001FF	All requests from iA Cores
IA_CLFLUSH	b00000001	0xC8C7FF	CLFlushes issued by iA Cores
IA_CLFLUSHOPT	b00000001	0xC8D7FF	CLFlushOpts issued by iA Cores
IA_CRD	b00000001	0xC80FFF	CRDs issued by iA Cores
IA_DRD	b00000001	0xC817FF	DRds issued by iA Cores
IA_DRDPTE	b00000001	0xC837FF	DRdPte issued by iA Cores due to a page walk
IA_DRD_OPT	b00000001	0xC827FF	DRd_Opts issued by iA Cores
IA_DRD_OPT_PREF	b00000001	0xC8A7FF	DRd_Opt_Prefs issued by iA Cores
IA_DRD_PREF	b00000001	0xC897FF	DRd_Prefs issued by iA Cores
IA_HIT	b00000001	0xC001FD	All requests from iA Cores that Hit the LLC
IA_HIT_CRD	b00000001	0xC80FFD	CRds issued by iA Cores that Hit the LLC
IA_HIT_CRD_PREF	b00000001	0xC88FFD	CRd_Prefs issued by iA Cores that hit the LLC
IA_HIT_DRD	b00000001	0xC817FD	DRds issued by iA Cores that Hit the LLC
IA_HIT_DRDPTE	b00000001	0xC837FD	DRdPte issued by iA Cores due to a page walk that hit the LLC
IA_HIT_DRD_OPT	b00000001	0xC827FD	DRd_Opts issued by iA Cores that hit the LLC
IA_HIT_DRD_OPT_PREF	b00000001	0xC8A7FD	DRd_Opt_Prefs issued by iA Cores that hit the LLC
IA_HIT_DRD_PREF	b00000001	0xC897FD	DRd_Prefs issued by iA Cores that Hit the LLC
IA_HIT_ITOM	b00000001	0xCC47FD	ItoMs issued by iA Cores that Hit LLC
IA_HIT_LLCPREFCODE	b00000001	0xCCCFFD	LLCPrefCode issued by iA Cores that hit the LLC
IA_HIT_LLCPREFDATA	b00000001	0xCCD7FD	LLCPrefData issued by iA Cores that hit the LLC
IA_HIT_LLCPREFRFO	b00000001	0xCCC7FD	LLCPrefRFO issued by iA Cores that hit the LLC
IA_HIT_RFO	b00000001	0xC807FD	RFOs issued by iA Cores that Hit the LLC
IA_HIT_RFO_PREF	b00000001	0xC887FD	RFO_Prefs issued by iA Cores that Hit the LLC
IA_ITOM	b00000001	0xCC47FF	ItoMs issued by iA Cores

Table 2-125. Unit Masks for TOR_OCCUPANCY (Sheet 3 of 6)

Extension	umask [15:8]	xtra [57:32]	Description
IA_ITOMCACHENEAR	b00000001	0xCD47FF	ItoMCacheNears issued by iA Cores
IA_LLCPREFCODE	b00000001	0xCCCFFF	LLCPrefCode issued by iA Cores
IA_LLCPREFDATA	b00000001	0xCCD7FF	LLCPrefData issued by iA Cores
IA_LLCPREFRFO	b00000001	0xCCC7FF	LLCPrefRFO issued by iA Cores
IA_MISS	b00000001	0xC001FE	All requests from iA Cores that Missed the LLC
IA_MISS_CRD	b00000001	0xC80FFE	CRds issued by iA Cores that Missed the LLC
IA_MISS_CRD_LOCAL	b00000001	0xC80EFE	CRd issued by iA Cores that Missed the LLC - HOMed locally
IA_MISS_CRD_PREF	b00000001	0xC88FFE	CRd_Prefs issued by iA Cores that Missed the LLC
IA_MISS_CRD_PREF_LOCAL	b00000001	0xC88EFE	CRd_Prefs issued by iA Cores that Missed the LLC - HOMed locally
IA_MISS_CRD_PREF_REMOTE	b00000001	0xC88F7E	CRd_Prefs issued by iA Cores that Missed the LLC - HOMed remotely
IA_MISS_CRD_REMOTE	b00000001	0xC80F7E	CRd issued by iA Cores that Missed the LLC - HOMed remotely
IA_MISS_DRD	b00000001	0xC817FE	DRds issued by iA Cores that Missed the LLC
IA_MISS_DRDPTE	b00000001	0xC837FE	DRdPte issued by iA Cores due to a page walk that missed the LLC
IA_MISS_DRD_DDR	b00000001	0xC81786	DRds issued by iA Cores targeting DDR Mem that Missed the LLC
IA_MISS_DRD_LOCAL	b00000001	0xC816FE	DRds issued by iA Cores that Missed the LLC - HOMed locally
IA_MISS_DRD_LOCAL_DDR	b00000001	0xC81686	DRds issued by iA Cores targeting DDR Mem that Missed the LLC - HOMed locally
IA_MISS_DRD_LOCAL_PMM	b00000001	0xC8168A	DRds issued by iA Cores targeting PMM Mem that Missed the LLC - HOMed locally
IA_MISS_DRD_OPT	b00000001	0xC827FE	DRd_Opt issued by iA Cores that missed the LLC
IA_MISS_DRD_OPT_PREF	b00000001	0xC8A7FE	DRd_Opt_Prefs issued by iA Cores that missed the LLC
IA_MISS_DRD_PMM	b00000001	0xC8178A	DRds issued by iA Cores targeting PMM Mem that Missed the LLC
IA_MISS_DRD_PREF	b00000001	0xC897FE	DRd_Prefs issued by iA Cores that Missed the LLC
IA_MISS_DRD_PREF_DDR	b00000001	0xC89786	DRd_Prefs issued by iA Cores targeting DDR Mem that Missed the LLC

Table 2-125. Unit Masks for TOR_OCCUPANCY (Sheet 4 of 6)

Extension	umask [15:8]	xtra [57:32]	Description
IA_MISS_DRD_PREF_LOCAL_DDR	b00000001	0xC89686	DRd_Prefs issued by iA Cores targeting DDR Mem that Missed the LLC - HOMed locally
IA_MISS_DRD_PREF_LOCAL_PMM	b00000001	0xC8968A	DRd_Prefs issued by iA Cores targeting PMM Mem that Missed the LLC - HOMed locally
IA_MISS_DRD_PREF_PMM	b00000001	0xC8978A	DRd_Prefs issued by iA Cores targeting PMM Mem that Missed the LLC
IA_MISS_DRD_PREF_REMOTE_DDR	b00000001	0xC89706	DRd_Prefs issued by iA Cores targeting DDR Mem that Missed the LLC - HOMed remotely
IA_MISS_DRD_PREF_REMOTE_PMM	b00000001	0xC8970A	DRd_Prefs issued by iA Cores targeting PMM Mem that Missed the LLC - HOMed remotely
IA_MISS_DRD_REMOTE	b00000001	0xC8177E	DRds issued by iA Cores that Missed the LLC - HOMed remotely
IA_MISS_DRD_REMOTE_DDR	b00000001	0xC81706	DRds issued by iA Cores targeting DDR Mem that Missed the LLC - HOMed remotely
IA_MISS_DRD_REMOTE_PMM	b00000001	0xC8170A	DRds issued by iA Cores targeting PMM Mem that Missed the LLC - HOMed remotely
IA_MISS_ITOM	b00000001	0xCC47FE	ItoMs issued by iA Cores that Missed LLC
IA_MISS_LLCPREFCODE	b00000001	0xCCCFFE	LLCPrefCode issued by iA Cores that missed the LLC
IA_MISS_LLCPREFDATA	b00000001	0xCCD7FE	LLCPrefData issued by iA Cores that missed the LLC
IA_MISS_LLCPREFRFO	b00000001	0xCCC7FE	LLCPrefRFO issued by iA Cores that missed the LLC
IA_MISS_LOCAL_WCILF_DDR	b00000001	0xC86686	WCiLFs issued by iA Cores targeting DDR that missed the LLC - HOMed locally
IA_MISS_LOCAL_WCILF_PMM	b00000001	0xC8668A	WCiLFs issued by iA Cores targeting PMM that missed the LLC - HOMed locally
IA_MISS_LOCAL_WCIL_DDR	b00000001	0xC86E86	WCiLs issued by iA Cores targeting DDR that missed the LLC - HOMed locally
IA_MISS_LOCAL_WCIL_PMM	b00000001	0xC86E8A	WCiLs issued by iA Cores targeting PMM that missed the LLC - HOMed locally
IA_MISS_REMOTE_WCILF_DDR	b00000001	0xC86706	WCiLFs issued by iA Cores targeting DDR that missed the LLC - HOMed remotely
IA_MISS_REMOTE_WCILF_PMM	b00000001	0xC8670A	WCiLFs issued by iA Cores targeting PMM that missed the LLC - HOMed remotely
IA_MISS_REMOTE_WCIL_DDR	b00000001	0xC86F06	WCiLs issued by iA Cores targeting DDR that missed the LLC - HOMed remotely
IA_MISS_REMOTE_WCIL_PMM	b00000001	0xC86F0A	WCiLs issued by iA Cores targeting PMM that missed the LLC - HOMed remotely

Table 2-125. Unit Masks for TOR_OCCUPANCY (Sheet 5 of 6)

Extension	umask [15:8]	xtra [57:32]	Description
IA_MISS_RFO	b00000001	0xC807FE	RFOs issued by iA Cores that Missed the LLC
IA_MISS_RFO_LOCAL	b00000001	0xC806FE	RFOs issued by iA Cores that Missed the LLC - HOMed locally
IA_MISS_RFO_PREF	b00000001	0xC887FE	RFO_Prefs issued by iA Cores that Missed the LLC
IA_MISS_RFO_PREF_LOCAL	b00000001	0xC886FE	RFO_Prefs issued by iA Cores that Missed the LLC - HOMed locally
IA_MISS_RFO_PREF_REMOTE	b00000001	0xC8877E	RFO_Prefs issued by iA Cores that Missed the LLC - HOMed remotely
IA_MISS_RFO_REMOTE	b00000001	0xC8077E	RFOs issued by iA Cores that Missed the LLC - HOMed remotely
IA_MISS_UCRDF	b00000001	0xC877DE	UCRdFs issued by iA Cores that Missed LLC
IA_MISS_WCIL	b00000001	0xC86FFE	WCiLs issued by iA Cores that Missed the LLC
IA_MISS_WCILF	b00000001	0xC867FE	WCiLF issued by iA Cores that Missed the LLC
IA_MISS_WCILF_DDR	b00000001	0xC86786	WCiLFs issued by iA Cores targeting DDR that missed the LLC
IA_MISS_WCILF_PMM	b00000001	0xC8678A	WCiLFs issued by iA Cores targeting PMM that missed the LLC
IA_MISS_WCIL_DDR	b00000001	0xC86F86	WCiLs issued by iA Cores targeting DDR that missed the LLC
IA_MISS_WCIL_PMM	b00000001	0xC86F8A	WCiLs issued by iA Cores targeting PMM that missed the LLC
IA_MISS_WIL	b00000001	0xC87FDE	WiLs issued by iA Cores that Missed LLC
IA_RFO	b00000001	0xC807FF	RFOs issued by iA Cores
IA_RFO_PREF	b00000001	0xC887FF	RFO_Prefs issued by iA Cores
IA_SPECITOM	b00000001	0xCC57FF	SpecItoMs issued by iA Cores
IA_WBMTOI	b00000001	0xCC27FF	WbMtoIs issued by iA Cores
IA_WCIL	b00000001	0xC86FFF	WCiLs issued by iA Cores
IA_WCILF	b00000001	0xC867FF	WCiLF issued by iA Cores
IRQ_IA	bxxxxxxx1	bxxxxxxxxxx xxxxxxxxxxx xxxxxxxxxx	IRQ - iA From an iA Core
LOC_IA	b00000001	0xC000FF	All from Local iA All locally initiated requests from iA Cores
EVICT	bxxxxxx1x	bxxxxxxxxxx xxxxxxxxxxx xxxxxxxxxx	SF/LLC Evictions TOR allocation occurred as a result of SF/LLC evictions (came from the ISMQ)
IO	b00000100	0xC001FF	All requests from IO Devices

Table 2-125. Unit Masks for TOR_OCCUPANCY (Sheet 6 of 6)

Extension	umask [15:8]	xtra [57:32]	Description
IO_CLFLUSH	b00000100	0xC8C3FF	CLFlushes issued by IO Devices
IO_HIT	b00000100	0xC001FD	All requests from IO Devices that hit the LLC
IO_HIT_ITOM	b00000100	0xCC43FD	ItoMs issued by IO Devices that Hit the LLC
IO_HIT_ITOMCACHENEAR	b00000100	0xCD43FD	ItoMCacheNears, indicating a partial write request, from IO Devices that hit the LLC
IO_HIT_PCIRDCUR	b00000100	0xC8F3FD	PCIRdCurs issued by IO Devices that hit the LLC
IO_HIT_RFO	b00000100	0xC803FD	RFOs issued by IO Devices that hit the LLC
IO_ITOM	b00000100	0xCC43FF	ItoMs issued by IO Devices
IO_ITOMCACHENEAR	b00000100	0xCD43FF	ItoMCacheNears, indicating a partial write request, from IO Devices
IO_MISS	b00000100	0xC001FE	All requests from IO Devices that missed the LLC
IO_MISS_ITOM	b00000100	0xCC43FE	ItoMs issued by IO Devices that missed the LLC
IO_MISS_ITOMCACHENEAR	b00000100	0xCD43FE	ItoMCacheNears, indicating a partial write request, from IO Devices that missed the LLC
IO_MISS_PCIRDCUR	b00000100	0xC8F3FE	PCIRdCurs issued by IO Devices that missed the LLC
IO_MISS_RFO	b00000100	0xC803FE	RFOs issued by IO Devices that missed the LLC
IO_PCIRDCUR	b00000100	0xC8F3FF	PCIRdCurs issued by IO Devices
IO_RFO	b00000100	0xC803FF	RFOs issued by IO Devices
IO_WBMTOI	b00000100	0xCC23FF	WbMtoIs issued by IO Devices
LOC_IO	b00000100	0xC000FF	All from Local IO All locally generated IO traffic
PRQ	bxxxxx1xx	bxxxxxxxxxx xxxxxxxxxxx xxxxxxxxxx	PRQ - IOSF From a PCIe Device
LOC_ALL	b00000101	0xC000FF	All from Local iA and IO All locally initiated requests
IPQ	bxxxx1xxx	bxxxxxxxxxx xxxxxxxxxxx xxxxxxxxxx	IPQ
IRQ_NON_IA	bxxx1xxxx	bxxxxxxxxxx xxxxxxxxxxx xxxxxxxxxx	IRQ - Non iA
PRQ_NON_IOSF	bxx1xxxxx	bxxxxxxxxxx xxxxxxxxxxx xxxxxxxxxx	PRQ - Non IOSF

WB_PUSH_MTOI

- **Title:**
- **Category:** HA WBPushMtoI Events
- **Event Code:** 0x56
- **Register Restrictions :** 0-3
- **Definition:** Counts the number of times when the CHA was received WbPushMtoI

Table 2-126. Unit Masks for WB_PUSH_MTOI

Extension	umask [15:8]	Description
LLC	bxxxxxxx1	Pushed to LLC Counts the number of times when the CHA was able to push WbPushMToI to LLC
MEM	bxxxxxx1x	Pushed to Memory Counts the number of times when the CHA was unable to push WbPushMToI to LLC (hence pushed it to MEM)

WRITE_NO_CREDITS

- **Title:**
- **Category:** MC Credit and Traffic Events
- **Event Code:** 0x5A
- **Register Restrictions :** 0-3
- **Definition:** Counts the number of times when there are no credits available for sending WRITEs from the CHA into the iMC. In order to send WRITEs into the memory controller, the HA must first acquire a credit for the iMCsBLIngressqueue.'

Table 2-127. Unit Masks for WRITE_NO_CREDITS

Extension	umask [15:8]	xtra [57:32]	Description
MC0	bxxxxxxx1	bxxxxxxxx	MC0 Filter for memory controller 0 only.
MC1	bxxxxxx1x	bxxxxxxxx	MC1 Filter for memory controller 1 only.
MC2	bxxxxx1xx	bxxxxxxxx	MC2 Filter for memory controller 2 only.
MC3	bxxxx1xxx	bxxxxxxxx	MC3 Filter for memory controller 3 only.
MC4	bxxx1xxxx	bxxxxxxxx	MC4 Filter for memory controller 4 only.
MC5	bxx1xxxxx	bxxxxxxxx	MC5 Filter for memory controller 5 only.

XPT_PREF

- **Title:**
- **Category:** XPT Events
- **Event Code:** 0x6F
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-128. Unit Masks for XPT_PREF

Extension	umask [15:8]	Description
SENT0	bxxxxxxx1	Sent (on 0?) Number of XPT prefetches sent
DROP0	bxxxxxx1x	Dropped (on 0?) Number of XPT prefetches dropped
DROP0_NOCRD	bxxxxx1xx	Dropped (on 0?) - No Credits Number of XPT prefetches dropped due to lack of XPT AD egress credits
DROP0_CONFLICT	bxxxx1xxx	Dropped (on 0?) - Conflict Number of XPT prefetches dropped due to AD CMS write port contention
SENT1	bxxx1xxxx	Sent (on 1?) Number of XPT prefetches sent
DROP1	bxx1xxxxx	Dropped (on 1?) Number of XPT prefetches dropped
DROP1_NOCRD	bx1xxxxxx	Dropped (on 1?) - No Credits Number of XPT prefetches dropped due to lack of XPT AD egress credits
DROP1_CONFLICT	b1xxxxxxx	Dropped (on 1?) - Conflict Number of XPT prefetches dropped due to AD CMS write port contention

2.3 Memory Controller (iMC) Performance Monitoring

The 5th Gen Intel® Xeon® Scalable Processor integrated Memory Controller provides the interface to DRAM and communicates to the rest of the Uncore through the Mesh2Mem block.

The memory controller also provides a variety of RAS features, such as ECC, memory access retry, memory scrubbing, thermal throttling, mirroring, and rank sparing.

2.3.1 Functional Overview

The memory controller communicates to DRAM, translating 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.

A selection of IMC functionality that performance monitoring provides some insight into:

- Supports up to 32 ranks per channel with 32 independent banks per rank.
- ECC support (correct any error within a x4 device)
- Open or closed page policy
- ISOCH
- Demand and Patrol Scrubbing support
- Support for LR-DIMMs (load reduced) for a buffered memory solution demanding higher capacity memory subsystems.

2.3.2 iMC Performance Monitoring Overview

The IMC supports event monitoring through four 48-bit wide counters (MC_CHy_PCI_PMON_CTR{3:0}) and one fixed counter (MC_CHy_PCI_PMON_FIXED_CTR) for each DRAM channel (of which there are two in the 5th Gen Intel® Xeon® Scalable Processor) the MC is attached to. Each of these counters can be programmed (MC_CHy_PCI_PMON_CTL{3:0}) to capture any MC event.

2.3.2.1 Additional iMC Performance Monitoring

Following is a counter that always tracks the number of DRAM clocks in the IMC. The DCLKS never changes frequency (on a given system), and therefore is a good measure of wall clock (unlike the Uncore clock which can change frequency based on system load).

Figure 2-5. PMON Control Register for DCLK

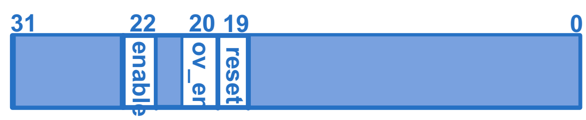


Table 2-129. MC_CHy_PCI_PMON_FIXED_CTL Register – Field Definitions

Field	Bits	Attr	HW Reset Val	Description
ig	31:23	RV	0	Ignored
en	22	RW-V	0	Local Counter Enable.
ig	21	RV	0	Ignored
ov_en	20	RW-V	0	When this bit is asserted and the corresponding counter overflows, a PMI exception is sent to the UBox.
rst	19	WO	0	When set to 1, the corresponding counter will be cleared to 0.
ig	18:0	RV	0	Ignored

Table 2-130. MC_CHy_PCI_PMON_CTR{FIXED,3-0} Register – Field Definitions

Field	Bits	Attr	HW Reset Val	Description
ig	63:48	RV	0	Ignored
event_count	47:0	RW-V	0	48-bit performance event counter

There are a few free-running counters, providing information highly valuable to a wide array of customers, in each iMC that collect counts for cumulative Read / Write Bandwidth across all channels.

'Free Running' counters cannot be changed by SW operating in a normal environment. SW cannot write them, cannot stop them and cannot reset the values.

Note: Counting will be suspended when the MC is powered down.

DDR CYCLES: There is one register per stack to track the number of DDR cycles as measured by MC.

Table 2-131. MC_MMIO_PMON_FRCTR_DCLK Register – Field Definitions

Field	Bits	Attr	HW Reset Val	Description
ig	63:48	RV	0	Ignored
event_count	47:0	RO-V	0	48-bit running count of DDR clocks captured in MC

WPQ ACTIVE CYCLES: counts the number of cycles WPQ was utilized over the total number of DDR cycles.

Table 2-132. MC_MMIO_PMON_FRCTR_WPQ_ACTIVE Register – Field Definitions

Field	Bits	Attr	HW Reset Val	Description
ig	63:48	RV	0	Ignored
event_count	47:0	RO-V	0	48-bit running count of data bytes read from attached DIMM

RPQ ACTIVE CYCLES: counts the number of cycles RPQ was utilized over the total number of DDR cycles.

Table 2-133. MC_MMIO_PMON_FRCTR_RPQ_ACTIVE Register – Field Definitions

Field	Bits	Attr	HW Reset Val	Description
ig	63:48	RV	0	Ignored
event_count	47:0	RO-V	0	48-bit running count of data bytes read from attached DIMM

2.3.3 iMC Box Events Ordered By Code

The following table summarizes the directly measured IMC Box events

Table 2-134. Directly Measured iMC Box Events (Sheet 1 of 3)

Symbol Name	Event Code	Ctrs	Description
CLOCKTICKS	0x1	0-3	IMC Clockticks at DCLK frequency
HCLOCKTICKS	0x1	0-3	IMC Clockticks at HCLK frequency
RPQ_INSERTS	0x10	0-3	Read Pending Queue Allocations
RPQ_CYCLES_NE	0x11	0-3	Read Pending Queue Not Empty
RPQ_CYCLES_FULL_PCH0	0x12	0-3	Read Pending Queue Full Cycles
RPQ_PRIO	0x13	0-3	
WPQ_PRIO	0x14	0-3	
RPQ_CYCLES_FULL_PCH1	0x15	0-3	Read Pending Queue Full Cycles
WPQ_CYCLES_FULL_PCH1	0x16	0-3	Write Pending Queue Full Cycles

Table 2-134. Directly Measured iMC Box Events (Sheet 2 of 3)

Symbol Name	Event Code	Ctrs	Description
RDB_INSERTS	0x17	0-3	Read Data Buffer Inserts
PREEMPTION_AS	0x1B	0-3	
PREEMPTION_MM_SWITCH	0x1C	0-3	
ACT_COUNT	0x2	0-3	DRAM Activate Count
WPQ_INSERTS	0x20	0-3	Write Pending Queue Allocations
WPQ_CYCLES_NE	0x21	0-3	Write Pending Queue Not Empty
WPQ_CYCLES_FULL_PCH0	0x22	0-3	Write Pending Queue Full Cycles
WPQ_READ_HIT	0x23	0-3	Write Pending Queue CAM Match
WPQ_WRITE_HIT	0x24	0-3	Write Pending Queue CAM Match
RD_CAS_PRIO	0x26	0-3	RD_CAS commands issued, by priority
WR_CAS_PRIO	0x27	0-3	WR_CAS commands issued, by priority
ACT_PRIO	0x28	0-3	ACT commands issued, by priority
CRIT_MINOR_CAS	0x2B	0-3	
PARITY_ERRORS	0x2C	0-3	
PRE_COUNT	0x3	0-3	DRAM Precharge commands.
CAS_MM	0x4	0-3	DRAM RD_CAS and WR_CAS Commands.
POWER_SELF_REFRESH	0x43	0-3	Clock-Enabled Self-Refresh
DRAM_PRE_ALL	0x44	0-3	DRAM Precharge All Commands
DRAM_REFRESH	0x45	0-3	Number of DRAM Refreshes Issued
POWER_THROTTLE_CYCLES	0x46	0-3	Throttle Cycles for Rank 0
POWER_CKE_CYCLES	0x47	0-3	CKE_ON_CYCLES by Rank
CAS_COUNT	0x5	0-3	DRAM RD_CAS and WR_CAS Commands.
MAJOR_MODE_CHANGE_PCH0	0x50	0-3	Major Mode Change PCH0
MAJOR_MODE_CHANGE_PCH1	0x51	0-3	Major Mode Change PCH1
MAJOR_MODE_VOTE_MISMATCH	0x52	0-3	
MAJOR_MODES	0x7	0-3	Cycles in a Major Mode
PREEMPTION	0x8	0-3	Read Preemption Count
RPQ_OCCUPANCY_PCH0	0x80	0-3	Read Pending Queue Occupancy
RPQ_OCCUPANCY_PCH1	0x81	0-3	Read Pending Queue Occupancy
WPQ_OCCUPANCY_PCH0	0x82	0-3	Write Pending Queue Occupancy
WPQ_OCCUPANCY_PCH1	0x83	0-3	Write Pending Queue Occupancy
POWER_CHANNEL_PPD	0x85	0-3	Channel PPD Cycles
POWER_CRIT_THROTTLE_CYCLES	0x86	0-3	Throttle Cycles for Rank 0
PCLS	0xA0	0-3	
SB_CYCLES_NE	0xD0	0-3	Scoreboard Cycles Not-Empty
SB_CYCLES_FULL	0xD1	0-3	Scoreboard Cycles Full

Table 2-134. Directly Measured iMC Box Events (Sheet 3 of 3)

Symbol Name	Event Code	Ctrs	Description
SB_ACCESSES	0xD2	0-3	Scoreboard Accesses
TAGCHK	0xD3	0-3	2LM Tag Check
SB_REJECT	0xD4	0-3	Number of Scoreboard Requests Rejected
SB_OCCUPANCY	0xD5	0-3	Scoreboard Occupancy
SB_INSERTS	0xD6	0-3	Scoreboard Inserts
SB_STRV_ALLOC	0xD7	0-3	Scoreboard allocations
SB_STRV_OCC	0xD8	0-3	Scoreboard occupancy
SB_CANARY	0xD9	0-3	
SB_PREF_INSERTS	0xDA	0-3	Scoreboard Prefetch Inserts
SB_PREF_OCCUPANCY	0xDB	0-3	Scoreboard Prefetch Occupancy
SB_TAGGED	0xDD	0-3	Scoreboard Tag
SB_STRV_DEALLOC	0xDE	0-3	
PMM_RPQ_OCCUPANCY	0xE0	0-3	PMM Read Pending Queue Occupancy
PMM_RPQ_CYCLES_NE	0xE1	0-3	PMM Read Queue Cycles Not Empty
PMM_RPQ_CYCLES_FULL	0xE2	0-3	PMM Read Queue Cycles Full
PMM_RPQ_INSERTS	0xE3	0-3	PMM Read Queue Inserts
PMM_WPQ_OCCUPANCY	0xE4	0-3	PMM Write Pending Queue Occupancy
PMM_WPQ_CYCLES_NE	0xE5	0-3	PMM Write Queue Cycles Not Empty
PMM_WPQ_CYCLES_FULL	0xE6	0-3	PMM Write Queue Cycles Full
PMM_WPQ_INSERTS	0xE7	0-3	PMM Write Queue Inserts
PMM_WPQ_FLUSH	0xE8	0-3	Write Pending Queue Flush
PMM_WPQ_FLUSH_CYC	0xE9	0-3	Write Pending Queue Flush cyscles
PMM_CMD1_SCH0	0xEA	0-3	PMM Commands
PMM_CMD1_SCH1	0xEB	0-3	PMM Commands - Part 2

2.3.4 iMC Box Common Metrics (Derived Events)

The following table summarizes metrics commonly calculated from IMC Box events

Table 2-135. Commonly Calculated From IMC Box Events (Sheet 1 of 2)

Symbol Name: Definition	Equation
MEM_BW_READS: Memory bandwidth consumed by reads. Expressed in bytes.	(CAS_COUNT.RD * 64)
MEM_BW_TOTAL: Total memory bandwidth. Expressed in bytes.	MEM_BW_READS + MEM_BW_WRITES
MEM_BW_WRITES: Memory bandwidth consumed by writes. Expressed in bytes.	(CAS_COUNT.WR * 64)

Table 2-135. Commonly Calculated From IMC Box Events (Sheet 2 of 2)

Symbol Name: Definition	Equation
PCT_CYCLES_CRITICAL_THROTTLE: The percentage of cycles all DRAM ranks in critical thermal throttling	POWER_CRITICAL_THROTTLE_CYCLES / MC_Chy_PCI_PMON_CTR_FIXED
PCT_CYCLES_DRAM_RANKx_IN_THR: The percentage of cycles DRAM rank (x) spent in thermal throttling.	POWER_THROTTLE_CYCLES.RANKx / MC_Chy_PCI_PMON_CTR_FIXED
PCT_CYCLES_PPD: The percentage of cycles Memory is in self refresh power mode	POWER_CHANNEL_PPD / MC_Chy_PCI_PMON_CTR_FIXED
PCT_CYCLES_SELF_REFRESH: The percentage of cycles Memory is in self refresh power mode	POWER_SELF_REFRESH / MC_Chy_PCI_PMON_CTR_FIXED
PCT_RD_REQUESTS: Percentage of read requests from total requests.	RPQ_INSERTS / (RPQ_INSERTS + WPQ_INSERTS)
PCT_WR_REQUESTS: Percentage of write requests from total requests.	WPQ_INSERTS / (RPQ_INSERTS + WPQ_INSERTS)

2.3.5 iMC Box Performance Monitor Event List

The section enumerates the 5th Gen Intel® Xeon® Scalable Processor performance monitoring events for the iMC Box.

ACT_COUNT

- **Title:**
- **Category:** ACT Events
- **Event Code:** 0x2
- **Register Restrictions :** 0-3
- **Definition:** Counts the number of DRAM Activate commands sent on this channel. Activate commands are issued to open up a page on the DRAM devices so that it can be read or written to with a CAS. One can calculate the number of page misses by subtracting the number of Page Miss precharges from the number of activates.

Table 2-136. Unit Masks for ACT_COUNT (Sheet 1 of 2)

Extension	umask [15:8]	Description
RD_PCH0	bxxxxxxx1	Activate due to Read in PCH0
WR_PCH0	bxxxxxx1x	Activate due to Write in PCH0
UFILL_PCH0	bxxxxx1xx	
RD_PCH1	bxxx1xxxx	Activate due to Read in PCH1
RD	bxxx1xxx1	Read transaction on Page Empty or Page Miss
WR_PCH1	bxx1xxxxx	Activate due to Write in PCH1
WR	bxx1xxx1x	Write transaction on Page Empty or Page Miss

Table 2-136. Unit Masks for ACT_COUNT (Sheet 2 of 2)

Extension	umask [15:8]	Description
UFILL_PCH1	bx1xxxxxx	
UFILL	bx1xxx1xx	Underfill Read transaction on Page Empty or Page Miss
ALL	b11111111	

ACT_PRIO

- **Title:**
- **Category:** CAS Events
- **Event Code:** 0x28
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-137. Unit Masks for ACT_PRIO

Extension	umask [15:8]	Description
PCH0	bx1xxxxxx	
PCH1	b1xxxxxxx	
RD_NORMAL	b11xxxxx1	Normal Reads
RD_CRITICAL	b11xxxx1x	Critical Reads
RD_STARVED	b11xxx1xx	Starved Reads
WR_NORMAL	b11xx1xxx	Normal Writes
WR_CRITICAL	b11x1xxxx	Critical Writes
WR_STARVED	b111xxxxx	Starved Writes

CAS_COUNT

- **Title:**
- **Category:** CAS Events
- **Event Code:** 0x5
- **Register Restrictions :** 0-3
- **Definition:** DRAM RD_CAS and WR_CAS commands.

Table 2-138. Unit Masks for CAS_COUNT

Extension	umask [15:8]	Description
PCH0	bx1xxxxxx	Pseudo Channel 0
PCH1	b1xxxxxxx	Pseudo Channel 1
RD_REG	b11xxxxx1	DRAM RD_CAS commands without auto-pre Counts the total number or DRAM Read CAS commands issued on this channel. This includes both regular RD CAS commands as well as those with implicit Precharge. We do not filter based on major mode, as RD_CAS is not issued during WMM (with the exception of underfill).
RD_PRE_REG	b11xxxx1x	DRAM RD_CAS commands w/auto-pre Counts the total number or DRAM Read CAS commands issued on this channel. This includes both regular RD CAS commands as well as those with explicit Precharge. AutoPre is only used in systems that are using closed page policy. We do not filter based on major mode, as RD_CAS is not issued during WMM (with the exception of underfills).
RD_UNDERFILL	b11xxx1xx	Underfill Read Issued
RD_PRE_UNDERFILL	b11xx1xxx	
RD	b11001111	All DRAM Reads Counts the total number of DRAM Read CAS commands, with and without auto-pre, issued on this channel. This includes underfills.
WR_NONPRE	b11x1xxxx	DRAM WR_CAS commands without auto-pre
WR_PRE	b111xxxxx	DRAM WR_CAS commands without auto-pre
WR	b11110000	All DRAM WR_CAS (both Modes) Counts the total number of DRAM Write CAS commands issued, with and without auto-pre, on this channel.
ALL	b11111111	All DRAM Read and Write actions Counts the total number of DRAM CAS commands issued on this channel.

CAS_MM

- **Title:**
- **Category:** Major Modes Events
- **Event Code:** 0x4
- **Register Restrictions :** 0-3
- **Definition:** DRAM RD_CAS and WR_CAS commands.

Table 2-139. Unit Masks for CAS_MM (Sheet 1 of 2)

Extension	umask [15:8]	Description
RD_RMM_PCH0	bxxxxxxx1	Read CAS issued in RMM
RD_WMM_PCH0	bxxxxxx1x	Read CAS issued in WMM
RD_RMM_PCH1	bxxxxx1xx	DRAM WR_CAS (with and without auto-pre) in Read Major Mode
RD_WMM_PCH1	bxxxx1xxx	DRAM WR_CAS (with and without auto-pre) in Write Major Mode

Table 2-139. Unit Masks for CAS_MM (Sheet 2 of 2)

Extension	umask [15:8]	Description
WR_RMM_PCH0	bxxx1xxxx	Read CAS issued in RMM
WR_WMM_PCH0	bxx1xxxxx	Read CAS issued in WMM
WR_RMM_PCH1	bx1xxxxxx	DRAM WR_CAS (with and without auto-pre) in Read Major Mode
WR_WMM_PCH1	b1xxxxxxx	DRAM WR_CAS (with and without auto-pre) in Write Major Mode

CLOCKTICKS

- **Title:**
- **Category:** DCLK Events
- **Event Code:** 0x1
- **Register Restrictions :** 0-3
- **Definition:**

CRIT_MINOR_CAS

- **Title:**
- **Category:** CAS Events
- **Event Code:** 0x2B
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-140. Unit Masks for CRIT_MINOR_CAS

Extension	umask [15:8]	Description
RD_PCH0	bxxxxxxx1	
RD_PCH1	bxxxxxx1x	
WR_PCH0	bxxxxx1xx	
WR_PCH1	bxxxx1xxx	

DRAM_PRE_ALL

- **Title:**
- **Category:** DRAM_PRE_ALL Events
- **Event Code:** 0x44
- **Register Restrictions :** 0-3
- **Definition:** Counts the number of times that the precharge all command was sent.

Table 2-141. Unit Masks for DRAM_PRE_ALL

Extension	umask [15:8]	Description
PCH0	bxxxxxxx1	DRAM Precharge -PCH0
PCH1	bxxxxxx1x	DRAM Precharge -PCH1

DRAM_REFRESH

- **Title:**
- **Category:** DRAM Refresh Events
- **Event Code:** 0x45
- **Register Restrictions :** 0-3
- **Definition:** Counts the number of refreshes issued.

Table 2-142. Unit Masks for DRAM_REFRESH

Extension	umask [15:8]	Description
PANIC_PCH0	bxxxxxx1x	Dram Refreshes - Panic PCH0
PANIC_PCH1	bxxx1xxxx	Dram Refreshes - Panic PCH1
PANIC_ALL	bxxx1xx1x	
HIGH_PCH1	bxx1xxxxx	Dram Refreshes - High PCH1
HIGH_ALL	bxx1xx1xx	

HCLOCKTICKS

- **Title:**
- **Category:** DCLK Events
- **Event Code:** 0x1
- **Register Restrictions :** 0-3
- **Definition:**

MAJOR_MODES

- **Title:**
- **Category:** Major Modes Events
- **Event Code:** 0x7
- **Register Restrictions :** 0-3
- **Definition:** Counts the total number of cycles spent in a major mode (selected by a filter) on the given channel. Major modes are channel-wide, and not a per-rank (or DIMM or bank) mode.

Table 2-143. Unit Masks for MAJOR_MODES

Extension	umask [15:8]	Description
PREF_RD	bxxxxxxx1	Preferred Reads
STARVED_RD	bxxxxxx1x	Starved Reads
PREF_WR	bxxxxx1xx	Preferred Writes
STARVED_WR	bxxxx1xxx	Starved Writes

MAJOR_MODE_CHANGE_PCH0

- **Title:**
- **Category:** Major Modes Events
- **Event Code:** 0x50
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-144. Unit Masks for MAJOR_MODE_CHANGE_PCH0

Extension	umask [15:8]	Description
PR_TO_SW	bxxxxxxx1	Ch 0 - Preferred Read to Starved Write
PR_TO_PW	bxxxxxx1x	Ch 0 - Preferred Read to Preferred Write
SW_TO_PR	bxxxxx1xx	Ch 0 - Starved Write to Preferred Read
SW_TO_PW	bxxxx1xxx	Ch 0 - Starved Write to Preferred Write
SR_TO_PR	bxxx1xxxx	Ch 0 - Starved Read to Preferred Read
SR_TO_PW	bxx1xxxxx	Ch 0 - Starved Read to Preferred write
PW_TO_SR	bx1xxxxxx	Ch 0 - Preferred Write to Starved Read
PW_TO_PR	b1xxxxxxx	Ch 0 - Preferred Write to Preferred Read

MAJOR_MODE_CHANGE_PCH1

- **Title:**
- **Category:** Major Modes Events
- **Event Code:** 0x51
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-145. Unit Masks for MAJOR_MODE_CHANGE_PCH1

Extension	umask [15:8]	Description
PR_TO_SW	bxxxxxxx1	Ch 1 - Preferred Read to Starved Write
PR_TO_PW	bxxxxxx1x	Ch 1 - Preferred Read to Preferred Write
SW_TO_PR	bxxxxx1xx	Ch 1 - Starved Write to Preferred Read
SW_TO_PW	bxxxx1xxx	Ch 1 - Starved Write to Preferred Write
SR_TO_PR	bxxx1xxxx	Ch 1 - Starved Read to Preferred Read
SR_TO_PW	bxx1xxxxx	Ch 1 - Starved Read to Preferred write
PW_TO_SR	bx1xxxxxx	Ch 1 - Preferred Write to Starved Read
PW_TO_PR	b1xxxxxxx	Ch 1 - Preferred Write to Preferred Read

MAJOR_MODE_VOTE_MISMATCH

- **Title:**
- **Category:** Major Modes Events
- **Event Code:** 0x52
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-146. Unit Masks for MAJOR_MODE_VOTE_MISMATCH

Extension	umask [15:8]	Description
PCH0	bxxxxxxx1	
PCH1	bxxxxxx1x	

PARITY_ERRORS

- **Title:**
- **Category:** Error Events
- **Event Code:** 0x2C
- **Register Restrictions :** 0-3
- **Definition:**

PCLS

- **Title:**
- **Category:** Debug Events
- **Event Code:** 0xA0
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-147. Unit Masks for PCLS

Extension	umask [15:8]	Description
RD_PCH0	bxxxxxxx1	PCLS read in PCH0
WR_PCH0	bxxxxxx1x	PCLS read in PCH0
RD_PCH1	bxxxxx1xx	PCLS write in PCH1
WR_PCH1	bxxxx1xxx	PCLS write in PCH1

PMM_CMD1_SCH0

- **Title:**
- **Category:** PMM CMD Events
- **Event Code:** 0xEA
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-148. Unit Masks for PMM_CMD1_SCH0

Extension	umask [15:8]	Description
ALL	bxxxxxxx1	All Counts all commands issued to PMM
RD	bxxxxxx1x	Reads - RPQ Counts read requests issued to the PMM RPQ
WR	bxxxxx1xx	Writes Counts write commands issued to PMM
UFILL_RD	bxxxx1xxx	Underfill reads Counts underfill read commands, due to a partial write, issued to PMM
GNT	bxxx1xxxx	RPQ GNTs
MISC	bxx1xxxxx	Underfill GNTs
OPP_RD	bx1xxxxxx	Misc GNTs
RD_REQS	b1xxxxxxx	Misc Commands (error, flow ACKs)

PMM_CMD1_SCH1

- **Title:**
- **Category:** PMM CMD Events
- **Event Code:** 0xEB
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-149. Unit Masks for PMM_CMD1_SCH1

Extension	umask [15:8]	Description
ALL	bxxxxxxx1	Opportunistic Reads
RD	bxxxxxx1x	Expected No data packet (ERID matched NDP encoding)
WR	bxxxxx1xx	Unexpected No data packet (ERID matched a Read, but data was a NDP)
UFILL_RD	bxxxx1xxx	Read Requests - Slot 0
GNT	bxxx1xxxx	Read Requests - Slot 1
MISC	bxx1xxxxx	ECC Errors
OPP_RD	bx1xxxxxx	ERID detectable parity error
RD_REQS	b1xxxxxxx	

PMM_RPQ_CYCLES_FULL

- **Title:**
- **Category:** PMM RPQ Events
- **Event Code:** 0xE2
- **Register Restrictions :** 0-3
- **Definition:**

PMM_RPQ_CYCLES_NE

- **Title:**
- **Category:** PMM RPQ Events
- **Event Code:** 0xE1
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-150. Unit Masks for PMM_RPQ_CYCLES_NE

Extension	umask [15:8]	Description
SCH0	bxxxxxxx1	
SCH1	bxxxxxx1x	

PMM_RPQ_INSERTS

- **Title:**
- **Category:** PMM RPQ Events
- **Event Code:** 0xE3
- **Register Restrictions :** 0-3

- **Definition:** Counts number of read requests allocated in the PMM Read Pending Queue. This includes both ISOCH and non-ISOCH requests.

PMM_RPQ_OCCUPANCY

- **Title:**
- **Category:** PMM RPQ Events
- **Event Code:** 0xE0
- **Register Restrictions :** 0-3
- **Definition:** Accumulates the per cycle occupancy of the PMM read pending queue.

Table 2-151. Unit Masks for PMM_RPQ_OCCUPANCY

Extension	umask [15:8]	Description
ALL_SCH0	bxxxxxxx1	
ALL_SCH1	bxxxxxx1x	
NO_GNT_SCH0	bxxxxx1xx	
NO_GNT_SCH1	bxxxx1xxx	
GNT_WAIT_SCH0	bxxx1xxxx	
GNT_WAIT_SCH1	bxx1xxxxx	

PMM_WPQ_CYCLES_FULL

- **Title:**
- **Category:** PMM WPQ Events
- **Event Code:** 0xE6
- **Register Restrictions :** 0-3
- **Definition:**

PMM_WPQ_CYCLES_NE

- **Title:**
- **Category:** PMM WPQ Events
- **Event Code:** 0xE5
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-152. Unit Masks for PMM_WPQ_CYCLES_NE

Extension	umask [15:8]	Description
SCH0	bxxxxxxx1	
SCH1	bxxxxxx1x	

PMM_WPQ_FLUSH

- **Title:**
- **Category:** PMM WPQ Events
- **Event Code:** 0xE8
- **Register Restrictions :** 0-3
- **Definition:**

PMM_WPQ_FLUSH_CYC

- **Title:**
- **Category:** PMM WPQ Events
- **Event Code:** 0xE9
- **Register Restrictions :** 0-3
- **Definition:**

PMM_WPQ_INSERTS

- **Title:**
- **Category:** PMM WPQ Events
- **Event Code:** 0xE7
- **Register Restrictions :** 0-3
- **Definition:** Counts number of write requests allocated in the PMM write pending queue.

PMM_WPQ_OCCUPANCY

- **Title:**
- **Category:** PMM WPQ Events
- **Event Code:** 0xE4
- **Register Restrictions :** 0-3
- **Definition:** Accumulates the per cycle occupancy of the PMM write pending queue.

Table 2-153. Unit Masks for PMM_WPQ_OCCUPANCY

Extension	umask [15:8]	Description
ALL_SCH0	bxxxxxxx1	
ALL_SCH1	bxxxxxx1x	
CAS_SCH0	bxxxxx1xx	
CAS_SCH1	bxxxx1xxx	
PWR_SCH0	bxxx1xxxx	
PWR_SCH1	bxx1xxxxx	

POWER_CHANNEL_PPD

- **Title:**
- **Category:** Power Events
- **Event Code:** 0x85
- **Register Restrictions :** 0-3
- **Definition:** The percentage of cycles DRAM rank (x) spent in thermal throttling.

POWER_CKE_CYCLES

- **Title:**
- **Category:** Power Events
- **Event Code:** 0x47
- **Register Restrictions :** 0-3
- **Definition:** Number of cycles spent in CKE ON mode. The filter allows you to select a rank to monitor. If multiple ranks are in CKE ON mode at one time, the counter will ONLY increment by one rather than doing accumulation. Multiple counters will need to be used to track multiple ranks simultaneously. There is no distinction between the different CKE modes (APD, PPDS, PPDF). This can be determined based on the system programming. These events should commonly be used with Invert to get the number of cycles in power saving mode. Edge Detect is also useful here. Make sure that you do NOT use invert with Edge Detect (this just confuses the system and is not necessary).

Table 2-154. Unit Masks for POWER_CKE_CYCLES

Extension	umask [15:8]	Description
LOW_0	b00000001	DIMM ID
LOW_1	b00000010	DIMM ID
LOW_2	b00000100	DIMM ID
LOW_3	b00001000	DIMM ID

POWER_CRIT_THROTTLE_CYCLES

- **Title:**
- **Category:** Power Events
- **Event Code:** 0x86
- **Register Restrictions :** 0-3
- **Definition:** Counts the number of cycles while the iMC is being throttled by either thermal constraints or by the PCU throttling. It is not possible to distinguish between the two. This can be filtered by rank. If multiple ranks are selected and are being throttled at the same time, the counter will only increment by 1.

Table 2-155. Unit Masks for POWER_CRIT_THROTTLE_CYCLES

Extension	umask [15:8]	Description
SLOT0	bxxxxxxx1	Slot0 Thermal throttling is performed per DIMM. We support 3 DIMMs per channel. This ID allows us to filter by ID.
SLOT1	bxxxxxx1x	Slot0

POWER_SELF_REFRESH

- **Title:**
- **Category:** Power Events
- **Event Code:** 0x43
- **Register Restrictions :** 0-3
- **Definition:** Counts the number of cycles when the iMC is in self-refresh and the iMC still has a clock. This happens in some package C-states. For example, the PCU may ask the iMC to enter self-refresh even though some of the cores are still processing. One use of this is for Monroe technology. Self-refresh is required during package C3 and C6, but there is no clock in the iMC at this time, so it is not possible to count these cases.

POWER_THROTTLE_CYCLES

- **Title:**
- **Category:** Power Events
- **Event Code:** 0x46
- **Register Restrictions :** 0-3
- **Definition:** Counts the number of cycles while the iMC is being throttled by either thermal constraints or by the PCU throttling. It is not possible to distinguish between the two. This can be filtered by rank. If multiple ranks are selected and are being throttled at the same time, the counter will only increment by 1.

Table 2-156. Unit Masks for POWER_THROTTLE_CYCLES

Extension	umask [15:8]	Description
SLOT0	bxxxxxxx1	Slot0 Thermal throttling is performed per DIMM. We support 3 DIMMs per channel. This ID allows us to filter by ID.
SLOT1	bxxxxxx1x	Slot1

PREEMPTION

- **Title:**
- **Category:** PREEMPTION Events
- **Event Code:** 0x8
- **Register Restrictions :** 0-3
- **Definition:** Counts the number of times a read in the iMC preempts another read or write. Generally reads to an open page are issued ahead of requests to closed pages. This improves the page hit rate of the system. However, high priority requests can cause pages of active requests to be closed in order to get them out. This will reduce the latency of the high-priority request at the expense of lower bandwidth and increased overall average latency.

Table 2-157. Unit Masks for PREEMPTION

Extension	umask [15:8]	Description
RD_PCH0	bxxxxxxx1	Ch 0 - Refresh over Read Preemption Filter for when a read preempts another read.
WR_PCH0	bxxxxxx1x	Ch 0 - Refresh over Write Preemption Filter for when a read preempts a write.
RD_PCH1	bxxxxx1xx	Ch 1 - Refresh over Read Preemption Filter for when a read preempts another read.
WR_PCH1	bxxxx1xxx	Ch 1 - Refresh over Write Preemption Filter for when a read preempts a write.

PREEMPTION_AS

- **Title:**
- **Category:** PREEMPTION Events
- **Event Code:** 0x1B
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-158. Unit Masks for PREEMPTION_AS

Extension	umask [15:8]	Description
PCH0	bxxxxxxx1	
PCH1	bxxxxxx1x	

PREEMPTION_MM_SWITCH

- **Title:**
- **Category:** PREEMPTION Events
- **Event Code:** 0x1C
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-159. Unit Masks for PREEMPTION_MM_SWITCH

Extension	umask [15:8]	Description
WMM_PCH0	bxxxxxxx1	Ch 0 - Write pre-empts Read that was in a BS
RMM_PCH0	bxxxxxx1x	Ch 0 - Read pre-empts Write that was in a BS
WMM_PCH1	bxxxxx1xx	Ch 1 - Write pre-empts Read that was in a BS
RMM_PCH1	bxxxx1xxx	Ch 1 - Read pre-empts Write that was in a BS

PRE_COUNT

- **Title:**
- **Category:** Precharge Events
- **Event Code:** 0x3
- **Register Restrictions :** 0-3
- **Definition:** Counts the number of DRAM precharge commands sent on this channel.

Table 2-160. Unit Masks for PRE_COUNT

Extension	umask [15:8]	Description
RD_PCH0	bxxxxxxx1	Precharge due to read - PCH0 Precharge from read bank scheduler
WR_PCH0	bxxxxxx1x	Precharge due to write - PCH0 Precharge from write bank scheduler
UFILL_PCH0	bxxxxx1xx	
PGT_PCH0	bxxxx1xxx	Precharges from Page Table Equivalent to PAGE_EMPTY
RD_PCH1	bxxx1xxxx	Precharge due to read - PCH1
RD	bxxx1xxx1	
WR_PCH1	bxx1xxxxx	Precharge due to write - PCH1
WR	bxx1xxx1x	
UFILL_PCH1	bx1xxxxxx	
UFILL	bx1xxx1xx	
PGT_PCH1	b1xxxxxxx	
PGT	b1xxx1xxx	
ALL	b11111111	

RDB_INSERTS

- **Title:**
- **Category:** RDB Events
- **Event Code:** 0x17
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-161. Unit Masks for RDB_INSERTS

Extension	umask [15:8]	Description
PCH0	bxxxxxxx1	Inserts in PCH0
PCH1	bxxxxxx1x	Inserts in PCH1

RD_CAS_PRIO

- **Title:**
- **Category:** CAS Events
- **Event Code:** 0x26
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-162. Unit Masks for RD_CAS_PRIO

Extension	umask [15:8]	Description
NORMAL_PCH0	bxxxxxxx1	Normal - PCH0
CRITICAL_PCH0	bxxxxxx1x	Critical - PCH0
STARVED_PCH0	bxxxxx1xx	Starved - PCH0
NORMAL_PCH1	bxxxx1xxx	Normal - PCH1
CRITICAL_PCH1	bxxx1xxxx	Critical - PCH1
STARVED_PCH1	bxx1Xxxxx	Starved - PCH1

RPQ_CYCLES_FULL_PCH0

- **Title:**
- **Category:** RPQ Events
- **Event Code:** 0x12
- **Register Restrictions :** 0-3
- **Definition:** Counts the number of cycles when the Read Pending Queue (RPQ) is full. When the RPQ is full, the HA will not be able to issue any additional read requests into the iMC. This count should be similar count in the HA which tracks the number of cycles that the HA has no RPQ credits, just somewhat smaller to account for the credit return overhead. We generally do not expect to see RPQ become full except for potentially during Write Major Mode or while running with slow DRAM. This event only tracks non-ISOC queue entries.

RPQ_CYCLES_FULL_PCH1

- **Title:**
- **Category:** RPQ Events
- **Event Code:** 0x15
- **Register Restrictions :** 0-3
- **Definition:** Counts the number of cycles when the RPQ is full. When the RPQ is full, the HA will not be able to issue any additional read requests into the iMC. This count should be similar count in the HA which tracks the number of cycles that the HA has no RPQ credits, just somewhat smaller to account for the credit return overhead. We generally do not expect to see RPQ become full except for potentially during Write

Major Mode or while running with slow DRAM. This event only tracks non-ISOC queue entries.

RPQ_CYCLES_NE

- **Title:**
- **Category:** RPQ Events
- **Event Code:** 0x11
- **Register Restrictions :** 0-3
- **Definition:** Counts the number of cycles that the RPQ is not empty. This can then be used to calculate the average occupancy (in conjunction with the Read Pending Queue Occupancy count). The RPQ is used to schedule reads out to the memory controller and to track the requests. Requests allocate into the RPQ soon after they enter the memory controller, and need credits for an entry in this buffer before being sent from the HA to the iMC. They deallocate after the CAS command has been issued to memory. This filter is to be used in conjunction with the occupancy filter so that one can correctly track the average occupancies for schedulable entries and scheduled requests.

Table 2-163. Unit Masks for RPQ_CYCLES_NE

Extension	umask [15:8]	Description
PCH0	bxxxxxxx1	Queue not empty in PCH0
PCH1	bxxxxxx1x	Queue not empty in PCH1

RPQ_INSERTS

- **Title:**
- **Category:** RPQ Events
- **Event Code:** 0x10
- **Register Restrictions :** 0-3
- **Definition:** Counts the number of allocations into the RPQ. This queue is used to schedule reads out to the memory controller and to track the requests. Requests allocate into the RPQ soon after they enter the memory controller, and need credits for an entry in this buffer before being sent from the HA to the iMC. They deallocate after the CAS command has been issued to memory. This includes both ISOCH and non-ISOCH requests.

Table 2-164. Unit Masks for RPQ_INSERTS

Extension	umask [15:8]	Description
PCH0	bxxxxxxx1	Inserts in PCH0
PCH1	bxxxxxx1x	Inserts in PCH1

RPQ_OCCUPANCY_PCH0

- **Title:**
- **Category:** RPQ Events
- **Event Code:** 0x80
- **Register Restrictions :** 0-3

- **Definition:** Accumulates the occupancies of the RPQ each cycle. This can then be used to calculate both the average occupancy (in conjunction with the number of cycles not empty) and the average latency (in conjunction with the number of allocations). The RPQ is used to schedule reads out to the memory controller and to track the requests. Requests allocate into the RPQ soon after they enter the memory controller, and need credits for an entry in this buffer before being sent from the HA to the iMC. They deallocate after the CAS command has been issued to memory.

RPQ_OCCUPANCY_PCH1

- **Title:**
- **Category:** RPQ Events
- **Event Code:** 0x81
- **Register Restrictions :** 0-3
- **Definition:** Accumulates the occupancies of the RPQ each cycle. This can then be used to calculate both the average occupancy (in conjunction with the number of cycles not empty) and the average latency (in conjunction with the number of allocations). The RPQ is used to schedule reads out to the memory controller and to track the requests. Requests allocate into the RPQ soon after they enter the memory controller, and need credits for an entry in this buffer before being sent from the HA to the iMC. They deallocate after the CAS command has been issued to memory.

RPQ_PRIO

- **Title:**
- **Category:** RPQ Events
- **Event Code:** 0x13
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-165. Unit Masks for RPQ_PRIO

Extension	umask [15:8]	Description
PCH0_LOW	bxxxxxxx1	
PCH0_MED	bxxxxxx1x	
PCH0_HIGH	bxxxxx1xx	
PCH0_CRIT	bxxxx1xxx	
PCH1_LOW	bxxx1xxxx	
PCH1_MED	bxx1xxxxx	
PCH1_HIGH	bx1xxxxxx	
PCH1_CRIT	b1xxxxxxx	

SB_ACCESSES

- **Title:**
- **Category:** PMM MEMMODE SCOREBOARD Events
- **Event Code:** 0xD2
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-166. Unit Masks for SB_ACCESSES

Extension	umask [15:8]	Description
RD_ACCEPTS	bxxxxxxx1	Reads Accepted
RD_REJECTS	bxxxxxx1x	Reads Rejected
WR_ACCEPTS	bxxxxx1xx	Writes Accepted
ACCEPTS	b00000101	Scoreboard Accesses Accepted
WR_REJECTS	bxxxx1xxx	Writes Rejected
REJECTS	b00001010	Scoreboard Accesses Rejected
NM_RD_CMPS	bxxx1xxxx	Near Mem read completions
NM_WR_CMPS	bxx1xxxxx	Near Mem write completions
FM_RD_CMPS	bx1xxxxxx	Far Mem read completions
FM_WR_CMPS	b1xxxxxxx	Far Mem write completions

SB_CANARY

- **Title:**
- **Category:** PMM MEMMODE SCOREBOARD Events
- **Event Code:** 0xD9
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-167. Unit Masks for SB_CANARY (Sheet 1 of 2)

Extension	umask [15:8]	Description
ALLOC	bxxxxxxx1	Alloc
DEALLOC	bxxxxxx1x	Dealloc
VLD	bxxxxx1xx	Valid
NM_RD_STARVED	bxxxx1xxx	Near Mem Reads Starved

Table 2-167. Unit Masks for SB_CANARY (Sheet 2 of 2)

Extension	umask [15:8]	Description
NM_WR_STARVED	bxxx1xxxx	Near Mem Writes Starved
FM_RD_STARVED	bxx1xxxxx	Far Mem Reads Starved
FM_WR_STARVED	bx1xxxxxx	Far Mem Writes Starved
FM_TGR_WR_STARVED	b1xxxxxxx	Far Mem TGR Writes Starved

SB_CYCLES_FULL

- **Title:**
- **Category:** PMM MEMMODE SCOREBOARD Events
- **Event Code:** 0xD1
- **Register Restrictions :** 0-3
- **Definition:**

SB_CYCLES_NE

- **Title:**
- **Category:** PMM MEMMODE SCOREBOARD Events
- **Event Code:** 0xD0
- **Register Restrictions :** 0-3
- **Definition:**

SB_INSERTS

- **Title:**
- **Category:** PMM MEMMODE SCOREBOARD Events
- **Event Code:** 0xD6
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-168. Unit Masks for SB_INSERTS (Sheet 1 of 2)

Extension	umask [15:8]	Description
RDS	bxxxxxxx1	Reads
WRS	bxxxxxx1x	Writes
PMM_RDS	bxxxxx1xx	Persistent Mem reads
PMM_WRS	bxxxx1xxx	Persistent Mem writes
BLOCK_RDS	bxxx1xxxx	Block region reads

Table 2-168. Unit Masks for SB_INSERTS (Sheet 2 of 2)

Extension	umask [15:8]	Description
BLOCK_WRS	bxx1xxxxx	Block region writes
PATROL	b1xxxxxxx	Patrol inserts

SB_OCCUPANCY

- **Title:**
- **Category:** PMM MEMMODE SCOREBOARD Events
- **Event Code:** 0xD5
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-169. Unit Masks for SB_OCCUPANCY

Extension	umask [15:8]	Description
RDS	bxxxxxxx1	Reads
WRS	bxxxxxx1x	Writes
PMM_RDS	bxxxxx1xx	Persistent Mem reads
PMM_WRS	bxxxx1xxx	Persistent Mem writes
BLOCK_RDS	bxx1xxxxx	Block region reads
BLOCK_WRS	bx1xxxxxx	Block region writes
PATROL	b1xxxxxxx	Patrol

SB_PREF_INSERTS

- **Title:**
- **Category:** PMM MEMMODE SCOREBOARD Events
- **Event Code:** 0xDA
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-170. Unit Masks for SB_PREF_INSERTS

Extension	umask [15:8]	Description
ALL	bxxxxxxx1	All
DDR	bxxxxxx1x	DDR4
PMM	bxxxxx1xx	Persistent Mem

SB_PREF_OCCUPANCY

- **Title:**
- **Category:** PMM MEMMODE SCOREBOARD Events
- **Event Code:** 0xDB
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-171. Unit Masks for SB_PREF_OCCUPANCY

Extension	umask [15:8]	Description
ALL	bxxxxxxx1	All
DDR	bxxxxxx1x	DDR4
PMM	bxxxxx1xx	Persistent Mem

SB_REJECT

- **Title:**
- **Category:** PMM MEMMODE COHERENCY Events
- **Event Code:** 0xD4
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-172. Unit Masks for SB_REJECT

Extension	umask [15:8]	Description
NM_SET_CNFLT	bxxxxxxx1	NM requests rejected due to set conflict
FM_ADDR_CNFLT	bxxxxxx1x	FM requests rejected due to full address conflict
PATROL_SET_CNFLT	bxxxxx1xx	Patrol requests rejected due to set conflict
CANARY	bxxxx1xxx	
DDR_EARLY_CMP	bxx1xxxxx	

SB_STRV_ALLOC

- **Title:**
- **Category:** PMM MEMMODE SCOREBOARD Events
- **Event Code:** 0xD7
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-173. Unit Masks for SB_STRV_ALLOC

Extension	umask [15:8]	Description
NM_RD	bxxxxxxx1	Near Mem Read - Set
FM_RD	bxxxxxx1x	Far Mem Read - Set
NM_WR	bxxxxx1xx	Near Mem Write - Set
FM_WR	bxxxx1xxx	Far Mem Write - Set
FM_TGR	bxxx1xxxx	Far Mem TGR

SB_STRV_DEALLOC

- **Title:**
- **Category:** PMM MEMMODE SCOREBOARD Events
- **Event Code:** 0xDE
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-174. Unit Masks for SB_STRV_DEALLOC

Extension	umask [15:8]	Description
NM_RD	bxxxxxxx1	Near Mem Read - Set
FM_RD	bxxxxxx1x	Far Mem Read - Set
NM_WR	bxxxxx1xx	Near Mem Write - Set
FM_WR	bxxxx1xxx	Far Mem Write - Set
FM_TGR	bxxx1xxxx	Far Mem TGR

SB_STRV_OCC

- **Title:**
- **Category:** PMM MEMMODE SCOREBOARD Events
- **Event Code:** 0xD8
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-175. Unit Masks for SB_STRV_OCC

Extension	umask [15:8]	Description
NM_RD	bxxxxxxx1	Near Mem Read
FM_RD	bxxxxxx1x	Far Mem Read
NM_WR	bxxxxx1xx	Near Mem Write
FM_WR	bxxxx1xxx	Far Mem Write
FM_TGR	bxxx1xxxx	Far Mem TGR

SB_TAGGED

- **Title:**
- **Category:** PMM MEMMODE SCOREBOARD Events
- **Event Code:** 0xDD
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-176. Unit Masks for SB_TAGGED

Extension	umask [15:8]	Description
NEW	bxxxxxxx1	
RD_HIT	bxxxxxx1x	
RD_MISS	bxxxxx1xx	
DDR4_CMP	bxxxx1xxx	
PMM0_CMP	bxxx1xxxx	
PMM1_CMP	bxx1xxxxx	
PMM2_CMP	bx1xxxxxx	
OCC	b1xxxxxxx	

WPQ_CYCLES_FULL_PCH0

- **Title:**
- **Category:** WPQ Events
- **Event Code:** 0x22
- **Register Restrictions :** 0-3
- **Definition:** Counts the number of cycles when the Write Pending Queue (WPQ) is full. When the WPQ is full, the HA will not be able to issue any additional write requests into the iMC. This count should be similar count in the CHA which tracks the

number of cycles that the CHA has no WPQ credits, just somewhat smaller to account for the credit return overhead.

WPQ_CYCLES_FULL_PCH1

- **Title:**
- **Category:** WPQ Events
- **Event Code:** 0x16
- **Register Restrictions :** 0-3
- **Definition:** Counts the number of cycles when the WPQ is full. When the WPQ is full, the HA will not be able to issue any additional write requests into the iMC. This count should be similar count in the CHA which tracks the number of cycles that the CHA has no WPQ credits, just somewhat smaller to account for the credit return overhead.

WPQ_CYCLES_NE

- **Title:**
- **Category:** WPQ Events
- **Event Code:** 0x21
- **Register Restrictions :** 0-3
- **Definition:** Counts the number of cycles that the WPQ is not empty. This can then be used to calculate the average queue occupancy (in conjunction with the WPQ Occupancy Accumulation count). The WPQ is used to schedule write out to the memory controller and to track the writes. Requests allocate into the WPQ soon after they enter the memory controller, and need credits for an entry in this buffer before being sent from the CHA to the iMC. They deallocate after being issued to DRAM. Write requests themselves are able to complete (from the perspective of the rest of the system) as soon they have "posted" to the iMC. This is not to be confused with actually performing the write to DRAM. Therefore, the average latency for this queue is actually not useful for deconstruction intermediate write latencies.

Table 2-177. Unit Masks for WPQ_CYCLES_NE

Extension	umask [15:8]	Description
PCH0	bxxxxxxx1	wpq not empty in PCH0
PCH1	bxxxxxx1x	wpq not empty in PCH1

WPQ_INSERTS

- **Title:**
- **Category:** WPQ Events
- **Event Code:** 0x20
- **Register Restrictions :** 0-3
- **Definition:** Counts the number of allocations into the WPQ. This can then be used to calculate the average queuing latency (in conjunction with the WPQ occupancy count). The WPQ is used to schedule write out to the memory controller and to track the writes. Requests allocate into the WPQ soon after they enter the memory controller, and need credits for an entry in this buffer before being sent from the CHA to the iMC. They deallocate after being issued to DRAM. Write requests themselves are able to complete (from the perspective of the rest of the system) as soon they have "posted" to the iMC.

Table 2-178. Unit Masks for WPQ_INSERTS

Extension	umask [15:8]	Description
PCH0	bxxxxxxx1	Inserts in PCH0
PCH1	bxxxxxx1x	Inserts in PCH1

WPQ_OCCUPANCY_PCH0

- **Title:**
- **Category:** WPQ Events
- **Event Code:** 0x82
- **Register Restrictions :** 0-3
- **Definition:** Accumulates the occupancies of the WPQ each cycle.This can then be used to calculate both the average queue occupancy (in conjunction with the number of cycles not empty) and the average latency (in conjunction with the number of allocations). The WPQ is used to schedule write out to the memory controller and to track the writes. Requests allocate into the WPQ soon after they enter the memory controller, and need credits for an entry in this buffer before being sent from the HA to the iMC. They deallocate after being issued to DRAM. Write requests themselves are able to complete (from the perspective of the rest of the system) as soon they have "posted" to the iMC. This is not to be confused with actually performing the write to DRAM. Therefore, the average latency for this queue is actually not useful for deconstruction intermediate write latencies.
  So, we provide filtering based on if the request has posted or not. By using the "not posted" filter, we can track how long writes spent in the iMC before completions were sent to the HA. The "posted" filter, on the other hand, provides information about how much queuing is actually happening in the iMC for writes before they are actually issued to memory. High average occupancies will generally coincide with high write major mode counts.

WPQ_OCCUPANCY_PCH1

- **Title:**
- **Category:** WPQ Events
- **Event Code:** 0x83
- **Register Restrictions :** 0-3
- **Definition:** Accumulates the occupancies of the WPQ each cycle. This can then be used to calculate both the average queue occupancy (in conjunction with the number of cycles not empty) and the average latency (in conjunction with the number of allocations). The WPQ is used to schedule write out to the memory controller and to track the writes. Requests allocate into the WPQ soon after they enter the memory controller, and need credits for an entry in this buffer before being sent from the HA to the iMC. They deallocate after being issued to DRAM. Write requests themselves are able to complete (from the perspective of the rest of the system) as soon they have "posted" to the iMC. This is not to be confused with actually performing the write to DRAM. Therefore, the average latency for this queue is actually not useful for deconstruction intermediate write latencies.
  So, we provide filtering based on if the request has posted or not. By using the "not posted" filter, we can track how long writes spent in the iMC before completions were sent to the HA. The "posted" filter, on the other hand, provides information about how much queuing is actually happening in the iMC for writes before they are actually issued to memory. High average occupancies will generally coincide with high write major mode counts.

WPQ_PRIO

- **Title:**
- **Category:** WPQ Events
- **Event Code:** 0x14
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-179. Unit Masks for WPQ_PRIO

Extension	umask [15:8]	Description
PCH0_LOW	bxxxxxxx1	
PCH0_MED	bxxxxxx1x	
PCH0_HIGH	bxxxxx1xx	
PCH0_CRIT	bxxxx1xxx	
PCH1_LOW	bxxx1xxxx	
PCH1_MED	bxx1xxxxx	
PCH1_HIGH	bx1xxxxxx	
PCH1_CRIT	b1xxxxxxx	

WPQ_READ_HIT

- **Title:**
- **Category:** WPQ Events
- **Event Code:** 0x23
- **Register Restrictions :** 0-3
- **Definition:** Counts the number of times a request hits in the WPQ. The iMC allows writes and reads to pass up other writes to different addresses. Before a read or a write is issued, it will first CAM the WPQ to see if there is a write pending to that address. When reads hit, they are able to directly pull their data from the WPQ instead of going to memory. Writes that hit will overwrite the existing data. Partial writes that hit will not need to do underfill reads and will simply update their relevant sections.

Table 2-180. Unit Masks for WPQ_READ_HIT

Extension	umask [15:8]	Description
PCH0	bxxxxxxx1	CAM Match in PCH0
PCH1	bxxxxxx1x	CAM Match in PCH1

WPQ_WRITE_HIT

- **Title:**
- **Category:** WPQ Events
- **Event Code:** 0x24
- **Register Restrictions :** 0-3
- **Definition:** Counts the number of times a request hits in the WPQ. The iMC allows writes and reads to pass up other writes to different addresses. Before a read or a write is issued, it will first CAM the WPQ to see if there is a write pending to that address. When reads hit, they are able to directly pull their data from the WPQ instead of going to memory. Writes that hit will overwrite the existing data. Partial writes that hit will not need to do Underfill reads and will simply update their relevant sections.

Table 2-181. Unit Masks for WPQ_WRITE_HIT

Extension	umask [15:8]	Description
PCH0	bxxxxxxx1	CAM Match in PCH0
PCH1	bxxxxxx1x	CAM Match in PCH1

WR_CAS_PRIO

- **Title:**
- **Category:** CAS Events
- **Event Code:** 0x27
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-182. Unit Masks for WR_CAS_PRIO

Extension	umask [15:8]	Description
NORMAL_PCH0	bxxxxxxx1	Normal - PCH0
CRITICAL_PCH0	bxxxxxx1x	Critical - PCH0
STARVED_PCH0	bxxxxx1xx	Starved - PCH0
NORMAL_PCH1	bxxxx1xxx	Normal - PCH1
CRITICAL_PCH1	bxxx1xxxx	Critical - PCH1
STARVED_PCH1	bxx1Xxxxx	Starved - PCH1

TAGCHK

- **Title:** 2LM Tag Check
- **Category:** TAG CHECK Events
- **Event Code:** 0xd3
- **Max. Inc/Cyc:** 1, **Register Restrictions:** 0-3
- **Definition:**

Table 2-183. Unit Masks for TAGCHK

Extension	umask [15:8]	Description
HIT	bxxxxxxx1	Hit in Near Memory Cache
MISS_CLEAN	bxxxxxx1x	Miss, no data in this line
MISS_DIRTY	bxxxxx1xx	Miss, existing data may be evicted to Far Memory
NM_RD_HIT	bxxxx1xxx	Read Hit in Near Memory Cache
NM_WR_HIT	bxxx1xxxx	Write Hit in Near Memory Cache

2.4 IIO Performance Monitoring

IIO stacks are responsible for managing traffic between the PCI Express* (PCIe*) domain and the mesh domain. The IIO PMON block is situated near the IIO stack's traffic controller capturing traffic controller as well as PCIe root port information. The traffic controller is responsible for translating traffic coming in from the Mesh and processed by IRP into the PCIe domain to IO agents such as CBDMA, DMA and PCIe.

2.4.1 IIO Performance Monitoring Overview

Each IIO Box, which sits near the IIO stack's Traffic Controller, supports event monitoring through four 48b wide counters (IIO{5-0}_MSR_PMON_CTR/CTL{3:0}). Each of these counters can be programmed to count any IIO event.

2.4.2 Additional IIO Performance Monitoring

2.4.2.1 IIO PMON Counter Control - Difference from Baseline

IIO performance monitoring control registers provide a small amount of additional functionality. The following table defines those cases.

Figure 2-6. IIO Counter Control Register for 5th Gen Intel® Xeon® Scalable Processor

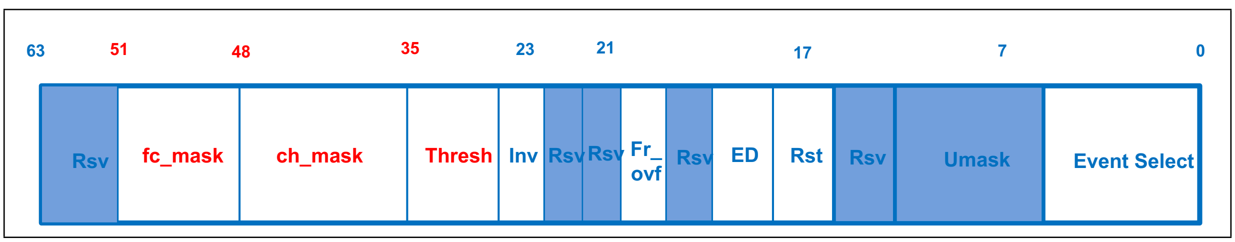


Table 2-184. IIOn_MSR_PMON_CTL{3-0} Register – Field Definitions

Field	Bits	Attr	HW Reset Val	Description
rsv	63:51	RV	0	Reserved. SW must write to 0 else behavior is undefined.
fc_mask	50:48	RW-V	0	FC Mask - applicable to certain events (Filter - fc) 0 - Posted Requests 1 - Non-posted Requests 2 - Completions
ch_mask	47:36	RW-V	0	Channel Mask Filter - applicable to certain events (Filter - channel)
thresh	35:24	RW-V	0	Threshold used in counter comparison.

There are a number of free-running counters, providing information highly valuable to a wide array of customers, in each IIO Stack that collect counts for input bandwidth for each Port.

'Free Running' counters cannot be changed by the SW operating in a normal environment. The SW cannot write them, cannot stop them and cannot reset the values.

Note: Counting will be suspended when the IIO stack is powered down.

There is one register per stack to track the number of IIO cycles.

Table 2-185. IIO_MSR_PMON_FRCTR_IOCLK Register – Field Definitions

Field	Bits	Attr	HW Reset Val	Description
ig	63:48	RV	0	Ignored
event_count	47:0	RO-V	0	48-bit running count of IO clocks.

- **Inbound (PCIe -> CPU) bandwidth** - counts DWs (4 bytes) of data, associated with writes and completions, transmitted from the IO stack to the traffic controller.

Table 2-186. IIO_MSR_PMON_FRCTR_BW_IN_P{0-7} Register – Field Definitions

Field	Bits	Attr	HW Reset Val	Description
ig	63:36	RV	0	Ignored
event_count	47:0	RO-V	0	48-bit running count of data bytes transmitted from link for this port.

- **Output (CPU -> PCIe) bandwidth** - counts DWs (4 bytes) of data, associated with writes and completions, transmitted from the traffic controller to the IO stack.

Table 2-187. IIO_MSR_PMON_FRCTR_BW_OUT_P{0-7} Register – Field Definitions

Field	Bits	Attr	HW Reset Val	Description
ig	63:36	RV	0	Ignored
event_count	47:0	RO-V	0	48-bit running count of data bytes transmitted link for this port.

2.4.3 IIO Performance Monitoring Events

IIO provides events to track information related to all the traffic passing through its boundaries.

- Bandwidth consumed and transactions processed broken down by transaction type.
- Per port utilization.
- Link power states.
- Completion buffer tracking.

2.4.4 IIO Box Events Ordered By Code

The following table summarizes the directly measured IIO Box events.

Table 2-188. Directly Measured IIO Box Events (Sheet 1 of 2)

Symbol Name	Event Code	Ctrs	Description
CLOCKTICKS	0x1	0-3	Traffic Controller Clocks
COMP_BUF_INSERTS	0xC2	0-3	PCIe Completion Buffer Inserts
DATA_REQ_OF_CPU	0x83	0-1	Data requested of the CPU
FASTPATH	0x89	0-3	
IOMMU0	0x40	0-3	IOMMU
IOMMU1	0x41	0-3	IOMMU
IOMMU3	0x43	0-3	IOMMU
MASK_MATCH_AND	0x2	0-1	AND Mask/match for debug bus
MASK_MATCH_OR	0x3	0-1	OR Mask/match for debug bus
NOTHING	0x0	0-3	Counting disabled
NUM_OUSTANDING_REQ_FROM_CPU	0xC5	2-3	Occupancy of outbound request queue
NUM_REQ_OF_CPU	0x85	0-3	Number requests PCIe makes of the main die
NUM_REQ_OF_CPU_BY_TGT	0x8E	0-3	Num requests sent by PCIe - by target
NUM_TGT_MATCHED_REQ_OF_CPU	0x8F	0-3	ITC address map 1
OUTBOUND_CL_REQS_ISSUED	0xD0	0-3	Outbound cacheline requests issued
OUTBOUND_TLP_REQS_ISSUED	0xD1	0-3	Outbound TLP (transaction layer packet) requests issued
PWT_OCCUPANCY	0x42	0-3	PWT occupancy
REQ_FROM_PCIE_CL_CMPL	0x91	0-3	One of the cache lines for a PCIE request is complete

Table 2-188. Directly Measured IIO Box Events (Sheet 2 of 2)

Symbol Name	Event Code	Ctrs	Description
REQ_FROM_PCIE_CMPL	0x92	0-3	PCIe Request complete
REQ_FROM_PCIE_PASS_CMPL	0x90	0-3	PCIe Request - pass complete
TXN_REQ_OF_CPU	0x84	0-3	Number Transactions requested of the CPU

2.4.5 IIO Box Performance Monitor Event List

The section enumerates 5th Gen Intel® Xeon® Scalable Processor performance monitoring events for the IIO Box.

CLOCKTICKS

- **Title:**
- **Category:** CLOCK Events
- **Event Code:** 0x1
- **Register Restrictions :** 0-3
- **Definition:**

COMP_BUF_INSERTS

- **Title:**
- **Category:** PCIe Completion Buffer Events
- **Event Code:** 0xC2
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-189. Unit Masks for COMP_BUF_INSERTS (Sheet 1 of 2)

Extension	umask [15:8]	xtra [50:36]	Description
CMPD.PART0	bxxxxx1xx	b111 bxxxxxxxxx xx1	Part 0 x16 card plugged in to stack, Or x8 card plugged in to Lane 0/1, Or x4 card is plugged in to slot 0
CMPD.PART1	bxxxxx1xx	b111 bxxxxxxxxx x1x	Part 1 x4 card is plugged in to slot 1
CMPD.PART2	bxxxxx1xx	b111 bxxxxxxxxx 1xx	Part 2 x8 card plugged in to Lane 2/3, Or x4 card is plugged in to slot 1
CMPD.PART3	bxxxxx1xx	b111 bxxxxxxxx1 xxx	Part 3 x4 card is plugged in to slot 3
CMPD.PART4	bxxxxx1xx	b111 bxxxxxxx1x xxx	Part 4 x16 card plugged in to stack, Or x8 card plugged in to Lane 0/1, Or x4 card is plugged in to slot 0
CMPD.PART5	bxxxxx1xx	b111 bxxxxxx1xx xxx	Part 5 x4 card is plugged in to slot 1

Table 2-189. Unit Masks for COMP_BUF_INSERTS (Sheet 2 of 2)

Extension	umask [15:8]	xtra [50:36]	Description
CMPD.PART6	bxxxxx1xx	b111 bxxxxx1xxx xxx	Part 6 x8 card plugged in to Lane 2/3, Or x4 card is plugged in to slot 1
CMPD.PART7	bxxxxx1xx	b111 bxxxx1xxxx xxx	Part 7 x4 card is plugged in to slot 3

DATA_REQ_OF_CPU

- **Title:**
- **Category:** Payload Events
- **Event Code:** 0x83
- **Register Restrictions :** 0-1
- **Definition:** Number of DWs (4 bytes) the card requests of the main die. Includes all requests initiated by the Card, including reads and writes.

Table 2-190. Unit Masks for DATA_REQ_OF_CPU (Sheet 1 of 5)

Extension	umask [15:8]	xtra [50:36]	Description
MEM_WRITE.IOMMU0	bxxxxxxx1	b111 bxxx1xxxxx xxx	Card writing to DRAM IOMMU - Type 0
MEM_WRITE.IOMMU1	bxxxxxxx1	b111 bxx1xxxxxx xxx	Card writing to DRAM IOMMU - Type 1
MEM_WRITE.PART0	bxxxxxxx1	b111 bxxxxxxxxx xx1	Card writing to DRAM x16 card plugged in to stack, Or x8 card plugged in to Lane 0/1, Or x4 card is plugged in to slot 0
MEM_WRITE.PART1	bxxxxxxx1	b111 bxxxxxxxxx x1x	Card writing to DRAM x4 card is plugged in to slot 1
MEM_WRITE.PART2	bxxxxxxx1	b111 bxxxxxxxxx 1xx	Card writing to DRAM x8 card plugged in to Lane 2/3, Or x4 card is plugged in to slot 1
MEM_WRITE.PART3	bxxxxxxx1	b111 bxxxxxxxx1 xxx	Card writing to DRAM x4 card is plugged in to slot 3
MEM_WRITE.PART4	bxxxxxxx1	b111 bxxxxxxx1x xxx	Card writing to DRAM x16 card plugged in to stack, Or x8 card plugged in to Lane 0/1, Or x4 card is plugged in to slot 0
MEM_WRITE.PART5	bxxxxxxx1	b111 bxxxxxx1xx xxx	Card writing to DRAM x4 card is plugged in to slot 1
MEM_WRITE.PART6	bxxxxxxx1	b111 bxxxxx1xxx xxx	Card writing to DRAM x8 card plugged in to Lane 2/3, Or x4 card is plugged in to slot 1
MEM_WRITE.PART7	bxxxxxxx1	b111 bxxxx1xxxx xxx	Card writing to DRAM x4 card is plugged in to slot 3
PEER_WRITE.IOMMU0	bxxxxxx1x	b111 bxxx1xxxxx xxx	Card writing to another Card (same or different stack) IOMMU - Type 0

Table 2-190. Unit Masks for DATA_REQ_OF_CPU (Sheet 2 of 5)

Extension	umask [15:8]	xtra [50:36]	Description
PEER_WRITE.IOMMU1	bxxxxxx1x	b111 bxx1xxxxxx xxx	Card writing to another Card (same or different stack) IOMMU - Type 1
PEER_WRITE.PART0	bxxxxxx1x	b111 bxxxxxxxxx xx1	Card writing to another Card (same or different stack) x16 card plugged in to stack, Or x8 card plugged in to Lane 0/1, Or x4 card is plugged in to slot 0
PEER_WRITE.PART1	bxxxxxx1x	b111 bxxxxxxxxx x1x	Card writing to another Card (same or different stack) x4 card is plugged in to slot 1
PEER_WRITE.PART2	bxxxxxx1x	b111 bxxxxxxxxx 1xx	Card writing to another Card (same or different stack) x8 card plugged in to Lane 2/3, Or x4 card is plugged in to slot 1
PEER_WRITE.PART3	bxxxxxx1x	b111 bxxxxxxxx1 xxx	Card writing to another Card (same or different stack) x4 card is plugged in to slot 3
PEER_WRITE.PART4	bxxxxxx1x	b111 bxxxxxxx1x xxx	Card writing to another Card (same or different stack) x16 card plugged in to stack, Or x8 card plugged in to Lane 0/1, Or x4 card is plugged in to slot 0
PEER_WRITE.PART5	bxxxxxx1x	b111 bxxxxxx1xx xxx	Card writing to another Card (same or different stack) x4 card is plugged in to slot 1
PEER_WRITE.PART6	bxxxxxx1x	b111 bxxxxx1xxx xxx	Card writing to another Card (same or different stack) x8 card plugged in to Lane 2/3, Or x4 card is plugged in to slot 1
PEER_WRITE.PART7	bxxxxxx1x	b111 bxxxx1xxxx xxx	Card writing to another Card (same or different stack) x4 card is plugged in to slot 3
MEM_READ.IOMMU0	bxxxxx1xx	b111 bxxx1xxxxx xxx	Card reading from DRAM IOMMU - Type 0
MEM_READ.IOMMU1	bxxxxx1xx	b111 bxx1xxxxxx xxx	Card reading from DRAM IOMMU - Type 1
MEM_READ.PART0	bxxxxx1xx	b111 bxxxxxxxxx xx1	Card reading from DRAM x16 card plugged in to stack, Or x8 card plugged in to Lane 0/1, Or x4 card is plugged in to slot 0
MEM_READ.PART1	bxxxxx1xx	b111 bxxxxxxxxx x1x	Card reading from DRAM x4 card is plugged in to slot 1
MEM_READ.PART2	bxxxxx1xx	b111 bxxxxxxxxx 1xx	Card reading from DRAM x8 card plugged in to Lane 2/3, Or x4 card is plugged in to slot 1
MEM_READ.PART3	bxxxxx1xx	b111 bxxxxxxxx1 xxx	Card reading from DRAM x4 card is plugged in to slot 3
MEM_READ.PART4	bxxxxx1xx	b111 bxxxxxxx1x xxx	Card reading from DRAM x16 card plugged in to stack, Or x8 card plugged in to Lane 0/1, Or x4 card is plugged in to slot 0
MEM_READ.PART5	bxxxxx1xx	b111 bxxxxxx1xx xxx	Card reading from DRAM x4 card is plugged in to slot 1
MEM_READ.PART6	bxxxxx1xx	b111 bxxxxx1xxx xxx	Card reading from DRAM x8 card plugged in to Lane 2/3, Or x4 card is plugged in to slot 1

Table 2-190. Unit Masks for DATA_REQ_OF_CPU (Sheet 3 of 5)

Extension	umask [15:8]	xtra [50:36]	Description
MEM_READ.PART7	bxxxxx1xx	b111 bxxxx1xxxx xxx	Card reading from DRAM x4 card is plugged in to slot 3
PEER_READ.IOMMU0	bxxxx1xxx	b111 bxxx1xxxxx xxx	Card reading from another Card (same or different stack) IOMMU - Type 0
PEER_READ.IOMMU1	bxxxx1xxx	b111 bxx1xxxxxx xxx	Card reading from another Card (same or different stack) IOMMU - Type 1
PEER_READ.PART0	bxxxx1xxx	b111 bxxxxxxxxx xx1	Card reading from another Card (same or different stack) x16 card plugged in to stack, Or x8 card plugged in to Lane 0/1, Or x4 card is plugged in to slot 0
PEER_READ.PART1	bxxxx1xxx	b111 bxxxxxxxxx x1x	Card reading from another Card (same or different stack) x4 card is plugged in to slot 1
PEER_READ.PART2	bxxxx1xxx	b111 bxxxxxxxxx 1xx	Card reading from another Card (same or different stack) x8 card plugged in to Lane 2/3, Or x4 card is plugged in to slot 1
PEER_READ.PART3	bxxxx1xxx	b111 bxxxxxxxx1 xxx	Card reading from another Card (same or different stack) x4 card is plugged in to slot 3
PEER_READ.PART4	bxxxx1xxx	b111 bxxxxxxx1x xxx	Card reading from another Card (same or different stack) x16 card plugged in to stack, Or x8 card plugged in to Lane 0/1, Or x4 card is plugged in to slot 0
PEER_READ.PART5	bxxxx1xxx	b111 bxxxxxx1xx xxx	Card reading from another Card (same or different stack) x4 card is plugged in to slot 1
PEER_READ.PART6	bxxxx1xxx	b111 bxxxxx1xxx xxx	Card reading from another Card (same or different stack) x8 card plugged in to Lane 2/3, Or x4 card is plugged in to slot 1
PEER_READ.PART7	bxxxx1xxx	b111 bxxxx1xxxx xxx	Card reading from another Card (same or different stack) x4 card is plugged in to slot 3
ATOMIC.IOMMU0	bxxx1xxxx	b111 bxxx1xxxxx xxx	Atomic requests targeting DRAM IOMMU - Type 0
ATOMIC.IOMMU1	bxxx1xxxx	b111 bxx1xxxxxx xxx	Atomic requests targeting DRAM IOMMU - Type 1
ATOMIC.PART0	bxxx1xxxx	b111 bxxxxxxxxx xx1	Atomic requests targeting DRAM x16 card plugged in to stack, Or x8 card plugged in to Lane 0/1, Or x4 card is plugged in to slot 0
ATOMIC.PART1	bxxx1xxxx	b111 bxxxxxxxxx x1x	Atomic requests targeting DRAM x4 card is plugged in to slot 1
ATOMIC.PART2	bxxx1xxxx	b111 bxxxxxxxxx 1xx	Atomic requests targeting DRAM x8 card plugged in to Lane 2/3, Or x4 card is plugged in to slot 1
ATOMIC.PART3	bxxx1xxxx	b111 bxxxxxxxx1 xxx	Atomic requests targeting DRAM x4 card is plugged in to slot 3

Table 2-190. Unit Masks for DATA_REQ_OF_CPU (Sheet 4 of 5)

Extension	umask [15:8]	xtra [50:36]	Description
ATOMIC.PART4	bxxx1xxxx	b111 bxxxxxxx1x xxx	Atomic requests targeting DRAM x16 card plugged in to stack, Or x8 card plugged in to Lane 0/1, Or x4 card is plugged in to slot 0
ATOMIC.PART5	bxxx1xxxx	b111 bxxxxxx1xx xxx	Atomic requests targeting DRAM x4 card is plugged in to slot 1
ATOMIC.PART6	bxxx1xxxx	b111 bxxxxx1xxx xxx	Atomic requests targeting DRAM x8 card plugged in to Lane 2/3, Or x4 card is plugged in to slot 1
ATOMIC.PART7	bxxx1xxxx	b111 bxxxx1xxxx xxx	Atomic requests targeting DRAM x4 card is plugged in to slot 3
MSG.IOMMU0	bx1xxxxxx	b111 bxxx1xxxxx xxx	Messages IOMMU - Type 0
MSG.IOMMU1	bx1xxxxxx	b111 bxx1xxxxxx xxx	Messages IOMMU - Type 1
MSG.PART0	bx1xxxxxx	b111 bxxxxxxxxx xx1	Messages x16 card plugged in to stack, Or x8 card plugged in to Lane 0/1, Or x4 card is plugged in to slot 0
MSG.PART1	bx1xxxxxx	b111 bxxxxxxxxx x1x	Messages x4 card is plugged in to slot 1
MSG.PART2	bx1xxxxxx	b111 bxxxxxxxxx 1xx	Messages x8 card plugged in to Lane 2/3, Or x4 card is plugged in to slot 1
MSG.PART3	bx1xxxxxx	b111 bxxxxxxxx1 xxx	Messages x4 card is plugged in to slot 3
MSG.PART4	bx1xxxxxx	b111 bxxxxxxx1x xxx	Messages x16 card plugged in to stack, Or x8 card plugged in to Lane 0/1, Or x4 card is plugged in to slot 0
MSG.PART5	bx1xxxxxx	b111 bxxxxxx1xx xxx	Messages x4 card is plugged in to slot 1
MSG.PART6	bx1xxxxxx	b111 bxxxxx1xxx xxx	Messages x8 card plugged in to Lane 2/3, Or x4 card is plugged in to slot 1
MSG.PART7	bx1xxxxxx	b111 bxxxx1xxxx xxx	Messages x4 card is plugged in to slot 3
CMPD.IOMMU0	b1xxxxxxx	b111 bxxx1xxxxx xxx	CmpD IOMMU - Type 0
CMPD.IOMMU1	b1xxxxxxx	b111 bxx1xxxxxx xxx	CmpD IOMMU - Type 1
CMPD.PART0	b1xxxxxxx	b111 bxxxxxxxxx xx1	CmpD x16 card plugged in to stack, Or x8 card plugged in to Lane 0/1, Or x4 card is plugged in to slot 0
CMPD.PART1	b1xxxxxxx	b111 bxxxxxxxxx x1x	CmpD x4 card is plugged in to slot 1

Table 2-190. Unit Masks for DATA_REQ_OF_CPU (Sheet 5 of 5)

Extension	umask [15:8]	xtra [50:36]	Description
CMPD.PART2	b1xxxxxxx	b111 bxxxxxxxxx 1xx	CmpD x8 card plugged in to Lane 2/3, Or x4 card is plugged in to slot 1
CMPD.PART3	b1xxxxxxx	b111 bxxxxxxxx1 xxx	CmpD x4 card is plugged in to slot 3
CMPD.PART4	b1xxxxxxx	b111 bxxxxxxx1x xxx	CmpD x16 card plugged in to stack, Or x8 card plugged in to Lane 0/1, Or x4 card is plugged in to slot 0
CMPD.PART5	b1xxxxxxx	b111 bxxxxxx1xx xxx	CmpD x4 card is plugged in to slot 1
CMPD.PART6	b1xxxxxxx	b111 bxxxxx1xxx xxx	CmpD x8 card plugged in to Lane 2/3, Or x4 card is plugged in to slot 1
CMPD.PART7	b1xxxxxxx	b111 bxxxx1xxxx xxx	CmpD x4 card is plugged in to slot 3

FASTPATH

- **Title:**
- **Category:** ITC Events
- **Event Code:** 0x89
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-191. Unit Masks for FASTPATH

Extension	umask [15:8]	xtra [50:36]	Description
REJECT	bxxxxxxx1	b11111111	FastPath Rejects
ACCEPT	bxxxxxx1x	b11111111	FastPath Accepts

IOMMU0

- **Title:**
- **Category:** IOMMU Events
- **Event Code:** 0x40
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-192. Unit Masks for IOMMU0

Extension	umask [15:8]	Description
FIRST_LOOKUPS	bxxxxxxx1	IOTLB lookups first Some transactions have to look up IOTLB multiple times. Counts the first time a request looks up IOTLB.
ALL_LOOKUPS	bxxxxxx1x	IOTLB lookups all Some transactions have to look up IOTLB multiple times. Counts every time a request looks up IOTLB.
4K_HITS	bxxxxx1xx	IOTLB Hits to a 4K Page Counts if a transaction to a 4K page, on its first lookup, hits the IOTLB.
2M_HITS	bxxxx1xxx	IOTLB Hits to a 2M Page Counts if a transaction to a 2M page, on its first lookup, hits the IOTLB.
1G_HITS	bxxx1xxxx	IOTLB Hits to a 1G Page Counts if a transaction to a 1G page, on its first lookup, hits the IOTLB.
MISSES	bxx1xxxxx	IOTLB Fills (same as IOTLB miss) When a transaction misses IOTLB, it does a page walk to look up memory and bring in the relevant page translation. Counts when this page translation is written to IOTLB.
CTXT_CACHE_LOOKUPS	bx1xxxxxx	Context cache lookups Counts each time a transaction looks up root context cache.
CTXT_CACHE_HITS	b1xxxxxxx	Context cache hits Counts each time a first look up of the transaction hits the RCC.

IOMMU1

- **Title:**
- **Category:** IOMMU Events
- **Event Code:** 0x41
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-193. Unit Masks for IOMMU1 (Sheet 1 of 2)

Extension	umask [15:8]	Description
SLPWC_CACHE_LOOKUPS	bxxxxxxx1	Second-Level Page Walk cache lookup Counts each time a transaction looks up second level page walk cache.
SLPWC_2M_HITS	bxxxxxx1x	SLPWC Hit to a 2M page Counts each time a transactions first lookup hits the SLPWC at the 2M level.'
SLPWC_1G_HITS	bxxxxx1xx	SLPWC Hit to a 1G page Counts each time a transaction’s first lookup hits the SLPWC at the 1G level.'
SLPWC_512G_HITS	bxxxx1xxx	SLPWC Hit to a 512G page Counts each time a transaction’s first lookup hits the SLPWC at the 512G level.'
SLPWC_256T_HITS	bxxx1xxxx	SLPWC Hit to a 256T page Counts each time a transaction’s first lookup hits the SLPWC at the 256T level.'

Table 2-193. Unit Masks for IOMMU1 (Sheet 2 of 2)

Extension	umask [15:8]	Description
SLPWC_CACHE_FILLS	bxx1xxxxx	Second-Level PageWalk cache fill When a transaction misses SLPWC, it does a page walk to look up memory and bring in the relevant page translation. Counts when this page translation is written to the SLPWC.
NUM_MEM_ACCESSES_LOW	bx1xxxxxx	IOMMU low priority memory access IOMMU sends out memory fetches when it misses the cache look up which is indicated by this signal. This indicates a low priority fetch.
NUM_MEM_ACCESSES_HIGH	b1xxxxxxx	IOMMU high priority memory access IOMMU sends out memory fetches when it misses the cache look up which is indicated by this signal. This indicates a high priority fetch.

IOMMU3

- **Title:**
- **Category:** IOMMU Events
- **Event Code:** 0x43
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-194. Unit Masks for IOMMU3

Extension	umask [15:8]	Description
PWT_OCCUPANCY_MSB	bxxxxxxx1	PWT occupancy MSB Additional bit for the page walks outstanding
CYC_PWT_FULL	bxxxxxx1x	Cycles PWT full Counts cycles the IOMMU has reached its maximum limit for outstanding page walks.
NUM_INVAL_IOTLB	bxxxxx1xx	IOTLB invalidation events Counts number of IOTLB invalidation events.
NUM_INVAL_CTXT_CACHE	bxxxx1xxx	Context Cache invalidation events Counts number of Context Cache invalidation events.
NUM_INVAL_PASID_CACHE	bxxx1xxxx	PASID Cache invalidation events Counts number of PASID Cache invalidation events.
NUM_INVAL_INT_CACHE	bxx1xxxxx	Interrupt Entry Cache invalidation events Counts number of Interrupt Entry Cache invalidation events.
INT_CACHE_LOOKUPS	bx1xxxxxx	Interrupt Entry cache lookup Counts the number of transaction lookups to the interrupt remapping cache.
INT_CACHE_HITS	b1xxxxxxx	Interrupt Entry cache hit Counts each time a transactions first lookup hits the IEC.

MASK_MATCH_AND

- **Title:**
- **Category:** Debug Events
- **Event Code:** 0x2
- **Register Restrictions :** 0-1
- **Definition:** Asserted if all bits specified by mask match.

Table 2-195. Unit Masks for MASK_MATCH_AND

Extension	umask [15:8]	Description
BUS0	bxxxxxxx1	Non-PCIE bus
BUS1	bxxxxxx1x	PCIE bus
BUS0_NOT_BUS1	bxxxxx1xx	Non-PCIE bus and !(PCIE bus)
BUS0_BUS1	bxxxx1xxx	Non-PCIE bus and PCIE bus
NOT_BUS0_BUS1	bxxx1xxxx	!(Non-PCIE bus) and PCIE bus
NOT_BUS0_NOT_BUS1	bxx1xxxxx	!(Non-PCIE bus) and !(PCIE bus)

MASK_MATCH_OR

- **Title:**
- **Category:** Debug Events
- **Event Code:** 0x3
- **Register Restrictions :** 0-1
- **Definition:** Asserted if any bits specified by mask match.

Table 2-196. Unit Masks for MASK_MATCH_OR

Extension	umask [15:8]	Description
BUS0	bxxxxxxx1	Non-PCIE bus
BUS1	bxxxxxx1x	PCIE bus
BUS0_NOT_BUS1	bxxxxx1xx	Non-PCIE bus and !(PCIE bus)
BUS0_BUS1	bxxxx1xxx	Non-PCIE bus and PCIE bus
NOT_BUS0_BUS1	bxxx1xxxx	!(Non-PCIE bus) and PCIE bus
NOT_BUS0_NOT_BUS1	bxx1xxxxx	!(Non-PCIE bus) and !(PCIE bus)

NOTHING

- **Title:**
- **Category:** CLOCK Events
- **Event Code:** 0x0
- **Register Restrictions :** 0-3
- **Definition:**

NUM_OUSTANDING_REQ_FROM_CPU

- **Title:**
- **Category:** OTC Events
- **Event Code:** 0xC5
- **Register Restrictions :** 2-3
- **Definition:** Counts number of outbound requests or completions the IIO is currently processing.

Table 2-197. Unit Masks for NUM_OUSTANDING_REQ_FROM_CPU

Extension	umask [15:8]	xtra [50:36]	Description
TO_HDR	bxxxxxx1x	b111 b11111111	To header stage
TO_IO	bxxxx1xxx	b111 b11111111	To device

NUM_REQ_OF_CPU_BY_TGT

- **Title:**
- **Category:** ITC Events
- **Event Code:** 0x8E
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-198. Unit Masks for NUM_REQ_OF_CPU_BY_TGT

Extension	umask [15:8]	xtra [50:36]	Description
MSGB	bxxxxxxx1	b111 b11111111	MsgB
MCAST	bxxxxxx1x	b111 b11111111	Multi-cast
UBOX	bxxxxx1xx	b111 b11111111	Ubox
MEM	bxxxx1xxx	b111 b11111111	Memory
REM_P2P	bxxx1xxxx	b111 b11111111	Remote P2P
LOC_P2P	bxx1xxxxx	b111 b11111111	Local P2P
CONFINED_P2P	bx1xxxxxx	b111 b11111111	Confined P2P
ABORT	b1xxxxxxx	b111 b11111111	Abort

NUM_TGT_MATCHED_REQ_OF_CPU

- **Title:**
- **Category:** ITC Events
- **Event Code:** 0x8F
- **Register Restrictions :** 0-3
- **Definition:**

OUTBOUND_CL_REQS_ISSUED

- **Title:**
- **Category:** OTC Events
- **Event Code:** 0xD0
- **Register Restrictions :** 0-3
- **Definition:** Each outbound cache line granular request may need to make multiple passes through the pipeline. Each time a cache line completes all its passes it advances line.

Table 2-199. Unit Masks for OUTBOUND_CL_REQS_ISSUED

Extension	umask [15:8]	xtra [50:36]	Description
TO_IO	bxxxx1xxx	b111 b11111111	64B requests issued to device

OUTBOUND_TLP_REQS_ISSUED

- **Title:**
- **Category:** OTC Events
- **Event Code:** 0xD1
- **Register Restrictions :** 0-3
- **Definition:** Each time an outbound completes all its passes it advances the pointer.

Table 2-200. Unit Masks for OUTBOUND_TLP_REQS_ISSUED

Extension	umask [15:8]	xtra [50:36]	Description
TO_IO	bxxxx1xxx	b111 b11111111	To device

PWT_OCCUPANCY

- **Title:**
- **Category:** IOMMU Events
- **Event Code:** 0x42
- **Register Restrictions :** 0-3
- **Definition:** Indicates how many page walks are outstanding at any point in time.

REQ_FROM_PCIE_CL_CMPL

- **Title:**
- **Category:** ITC Events
- **Event Code:** 0x91
- **Register Restrictions :** 0-3
- **Definition:** Each PCIe request is broken down into a series of cacheline granular requests and each cacheline size request may need to make multiple passes through the pipeline (for example, for posted interrupts or multi-cast). Each time a cacheline completes all its passes (for example, finishes posting writes to all multi-cast targets) it advances line.

Table 2-201. Unit Masks for REQ_FROM_PCIE_CL_CMPL

Extension	umask [15:8]	xtra [50:36]	Description
REQ_OWN	bxxxxx1xx	b111 b11111111	Request Ownership
FINAL_RD_WR	bxxxx1xxx	b111 b11111111	Issuing final read or write of line
WR	bxxx1xxxx	b111 b11111111	Writing line
DATA	bxx1xxxxx	b111 b11111111	Passing data to be written
NP	bx1xxxxxx	b111 b11111111	PCIe Request complete

REQ_FROM_PCIE_CMPL

- **Title:**
- **Category:** ITC Events
- **Event Code:** 0x92
- **Register Restrictions :** 0-3
- **Definition:** Each PCIe request is broken down into a series of cacheline granular requests and each cacheline size request may need to make multiple passes through the pipeline (for example, for posted interrupts or multi-cast). Each time a single PCIe request completes all its cacheline granular requests, it advances pointer.

Table 2-202. Unit Masks for REQ_FROM_PCIE_CMPL

Extension	umask [15:8]	xtra [50:36]	Description
IOMMU_REQ	bxxxxxxx1	b111 b11111111	Issuing to IOMMU
IOMMU_HIT	bxxxxxx1x	b111 b11111111	Processing response from IOMMU
REQ_OWN	bxxxxx1xx	b111 b11111111	Request Ownership
FINAL_RD_WR	bxxxx1xxx	b111 b11111111	Issuing final read or write of line

REQ_FROM_PCIE_PASS_CMPL

- **Title:**
- **Category:** ITC Events
- **Event Code:** 0x90
- **Register Restrictions :** 0-3
- **Definition:** Each PCIe request is broken down into a series of cache line granular requests and each cacheline size request may need to make multiple passes through the pipeline (for example, for posted interrupts or multi-cast). Each time a cache line completes a single pass (for example, posts a write to single multi-cast target) it advances state.

Table 2-203. Unit Masks for REQ_FROM_PCIE_PASS_CMPL

Extension	umask [15:8]	xtra [50:36]	Description
REQ_OWN	bxxxxx1xx	b111 b11111111	Request Ownership
FINAL_RD_WR	bxxxx1xxx	b111 b11111111	Issuing final read or write of line
WR	bxxx1xxxx	b111 b11111111	Writing line
DATA	bxx1xxxxx	b111 b11111111	Passing data to be written
NP	bx1xxxxxx	b111 b11111111	PCIe Request - cacheline complete

TXN_REQ_OF_CPU

- **Title:**
- **Category:** Transaction Events
- **Event Code:** 0x84
- **Register Restrictions :** 0-3
- **Definition:** Also known as inbound. Number of 64B cache line requests initiated by the card, including reads and writes.

Table 2-204. Unit Masks for TXN_REQ_OF_CPU (Sheet 1 of 5)

Extension	umask [15:8]	xtra [50:36]	Description
MEM_WRITE.IOMMU0	bxxxxxxx1	b111 bxxx1xxxxx xxx	Card writing to DRAM IOMMU - Type 0
MEM_WRITE.IOMMU1	bxxxxxxx1	b111 bxx1xxxxxx xxx	Card writing to DRAM IOMMU - Type 1
MEM_WRITE.PART0	bxxxxxxx1	b111 bxxxxxxxxx xx1	Card writing to DRAM x16 card plugged in to stack, Or x8 card plugged in to Lane 0/1, Or x4 card is plugged in to slot 0
MEM_WRITE.PART1	bxxxxxxx1	b111 bxxxxxxxxx x1x	Card writing to DRAM x4 card is plugged in to slot 1
MEM_WRITE.PART2	bxxxxxxx1	b111 bxxxxxxxxx 1xx	Card writing to DRAM x8 card plugged in to Lane 2/3, Or x4 card is plugged in to slot 1
MEM_WRITE.PART3	bxxxxxxx1	b111 bxxxxxxxx1 xxx	Card writing to DRAM x4 card is plugged in to slot 3
MEM_WRITE.PART4	bxxxxxxx1	b111 bxxxxxxx1x xxx	Card writing to DRAM x16 card plugged in to stack, Or x8 card plugged in to Lane 0/1, Or x4 card is plugged in to slot 0
MEM_WRITE.PART5	bxxxxxxx1	b111 bxxxxxx1xx xxx	Card writing to DRAM x4 card is plugged in to slot 1
MEM_WRITE.PART6	bxxxxxxx1	b111 bxxxxx1xxx xxx	Card writing to DRAM x8 card plugged in to Lane 2/3, Or x4 card is plugged in to slot 1

Table 2-204. Unit Masks for TXN_REQ_OF_CPU (Sheet 2 of 5)

Extension	umask [15:8]	xtra [50:36]	Description
MEM_WRITE.PART7	bxxxxxxx1	b111 bxxxx1xxxx xxx	Card writing to DRAM x4 card is plugged in to slot 3
PEER_WRITE.IOMMU0	bxxxxxx1x	b111 bxxx1xxxxx xxx	Card writing to another Card (same or different stack) IOMMU - Type 0
PEER_WRITE.IOMMU1	bxxxxxx1x	b111 bxx1xxxxxx xxx	Card writing to another Card (same or different stack) IOMMU - Type 1
PEER_WRITE.PART0	bxxxxxx1x	b111 bxxxxxxxxx xx1	Card writing to another Card (same or different stack) x16 card plugged in to stack, Or x8 card plugged in to Lane 0/1, Or x4 card is plugged in to slot 0
PEER_WRITE.PART1	bxxxxxx1x	b111 bxxxxxxxxx x1x	Card writing to another Card (same or different stack) x4 card is plugged in to slot 1
PEER_WRITE.PART2	bxxxxxx1x	b111 bxxxxxxxxx 1xx	Card writing to another Card (same or different stack) x8 card plugged in to Lane 2/3, Or x4 card is plugged in to slot 1
PEER_WRITE.PART3	bxxxxxx1x	b111 bxxxxxxxx1 xxx	Card writing to another Card (same or different stack) x4 card is plugged in to slot 3
PEER_WRITE.PART4	bxxxxxx1x	b111 bxxxxxxx1x xxx	Card writing to another Card (same or different stack) x16 card plugged in to stack, Or x8 card plugged in to Lane 0/1, Or x4 card is plugged in to slot 0
PEER_WRITE.PART5	bxxxxxx1x	b111 bxxxxxx1xx xxx	Card writing to another Card (same or different stack) x4 card is plugged in to slot 1
PEER_WRITE.PART6	bxxxxxx1x	b111 bxxxxx1xxx xxx	Card writing to another Card (same or different stack) x8 card plugged in to Lane 2/3, Or x4 card is plugged in to slot 1
PEER_WRITE.PART7	bxxxxxx1x	b111 bxxxx1xxxx xxx	Card writing to another Card (same or different stack) x4 card is plugged in to slot 3
MEM_READ.IOMMU0	bxxxxx1xx	b111 bxxx1xxxxx xxx	Card reading from DRAM IOMMU - Type 0
MEM_READ.IOMMU1	bxxxxx1xx	b111 bxx1xxxxxx xxx	Card reading from DRAM IOMMU - Type 1
MEM_READ.PART0	bxxxxx1xx	b111 bxxxxxxxxx xx1	Card reading from DRAM x16 card plugged in to stack, Or x8 card plugged in to Lane 0/1, Or x4 card is plugged in to slot 0
MEM_READ.PART1	bxxxxx1xx	b111 bxxxxxxxxx x1x	Card reading from DRAM x4 card is plugged in to slot 1
MEM_READ.PART2	bxxxxx1xx	b111 bxxxxxxxxx 1xx	Card reading from DRAM x8 card plugged in to Lane 2/3, Or x4 card is plugged in to slot 1
MEM_READ.PART3	bxxxxx1xx	b111 bxxxxxxxx1 xxx	Card reading from DRAM x4 card is plugged in to slot 3
MEM_READ.PART4	bxxxxx1xx	b111 bxxxxxxx1x xxx	Card reading from DRAM x16 card plugged in to stack, Or x8 card plugged in to Lane 0/1, Or x4 card is plugged in to slot 0

Table 2-204. Unit Masks for TXN_REQ_OF_CPU (Sheet 3 of 5)

Extension	umask [15:8]	xtra [50:36]	Description
MEM_READ.PART5	bxxxxx1xx	b111 bxxxxxx1xx xxx	Card reading from DRAM x4 card is plugged in to slot 1
MEM_READ.PART6	bxxxxx1xx	b111 bxxxxx1xxx xxx	Card reading from DRAM x8 card plugged in to Lane 2/3, Or x4 card is plugged in to slot 1
MEM_READ.PART7	bxxxxx1xx	b111 bxxxx1xxxx xxx	Card reading from DRAM x4 card is plugged in to slot 3
PEER_READ.IOMMU0	bxxxx1xxx	b111 bxxx1xxxxx xxx	Card reading from another Card (same or different stack) IOMMU - Type 0
PEER_READ.IOMMU1	bxxxx1xxx	b111 bxx1xxxxxx xxx	Card reading from another Card (same or different stack) IOMMU - Type 1
PEER_READ.PART0	bxxxx1xxx	b111 bxxxxxxxxx xx1	Card reading from another Card (same or different stack) x16 card plugged in to stack, Or x8 card plugged in to Lane 0/1, Or x4 card is plugged in to slot 0
PEER_READ.PART1	bxxxx1xxx	b111 bxxxxxxxxx x1x	Card reading from another Card (same or different stack) x4 card is plugged in to slot 1
PEER_READ.PART2	bxxxx1xxx	b111 bxxxxxxxxx 1xx	Card reading from another Card (same or different stack) x8 card plugged in to Lane 2/3, Or x4 card is plugged in to slot 1
PEER_READ.PART3	bxxxx1xxx	b111 bxxxxxxxx1 xxx	Card reading from another Card (same or different stack) x4 card is plugged in to slot 3
PEER_READ.PART4	bxxxx1xxx	b111 bxxxxxxx1x xxx	Card reading from another Card (same or different stack) x16 card plugged in to stack, Or x8 card plugged in to Lane 0/1, Or x4 card is plugged in to slot 0
PEER_READ.PART5	bxxxx1xxx	b111 bxxxxxx1xx xxx	Card reading from another Card (same or different stack) x4 card is plugged in to slot 1
PEER_READ.PART6	bxxxx1xxx	b111 bxxxxx1xxx xxx	Card reading from another Card (same or different stack) x8 card plugged in to Lane 2/3, Or x4 card is plugged in to slot 1
PEER_READ.PART7	bxxxx1xxx	b111 bxxxx1xxxx xxx	Card reading from another Card (same or different stack) x4 card is plugged in to slot 3
ATOMIC.IOMMU0	bxxx1xxxx	b111 bxxx1xxxxx xxx	Atomic requests targeting DRAM IOMMU - Type 0
ATOMIC.IOMMU1	bxxx1xxxx	b111 bxx1xxxxxx xxx	Atomic requests targeting DRAM IOMMU - Type 1
ATOMIC.PART0	bxxx1xxxx	b111 bxxxxxxxxx xx1	Atomic requests targeting DRAM x16 card plugged in to stack, Or x8 card plugged in to Lane 0/1, Or x4 card is plugged in to slot 0
ATOMIC.PART1	bxxx1xxxx	b111 bxxxxxxxxx x1x	Atomic requests targeting DRAM x4 card is plugged in to slot 1

Table 2-204. Unit Masks for TXN_REQ_OF_CPU (Sheet 4 of 5)

Extension	umask [15:8]	xtra [50:36]	Description
ATOMIC.PART2	bxxx1xxxx	b111 bxxxxxxxxx 1xx	Atomic requests targeting DRAM x8 card plugged in to Lane 2/3, Or x4 card is plugged in to slot 1
ATOMIC.PART3	bxxx1xxxx	b111 bxxxxxxxx1 xxx	Atomic requests targeting DRAM x4 card is plugged in to slot 3
ATOMIC.PART4	bxxx1xxxx	b111 bxxxxxxx1x xxx	Atomic requests targeting DRAM x16 card plugged in to stack, Or x8 card plugged in to Lane 0/1, Or x4 card is plugged in to slot 0
ATOMIC.PART5	bxxx1xxxx	b111 bxxxxxx1xx xxx	Atomic requests targeting DRAM x4 card is plugged in to slot 1
ATOMIC.PART6	bxxx1xxxx	b111 bxxxxx1xxx xxx	Atomic requests targeting DRAM x8 card plugged in to Lane 2/3, Or x4 card is plugged in to slot 1
ATOMIC.PART7	bxxx1xxxx	b111 bxxxx1xxxx xxx	Atomic requests targeting DRAM x4 card is plugged in to slot 3
MSG.IOMMU0	bx1xxxxxx	b111 bxxx1xxxxx xxx	Messages IOMMU - Type 0
MSG.IOMMU1	bx1xxxxxx	b111 bxx1xxxxxx xxx	Messages IOMMU - Type 1
MSG.PART0	bx1xxxxxx	b111 bxxxxxxxxx xx1	Messages x16 card plugged in to stack, Or x8 card plugged in to Lane 0/1, Or x4 card is plugged in to slot 0
MSG.PART1	bx1xxxxxx	b111 bxxxxxxxxx x1x	Messages x4 card is plugged in to slot 1
MSG.PART2	bx1xxxxxx	b111 bxxxxxxxxx 1xx	Messages x8 card plugged in to Lane 2/3, Or x4 card is plugged in to slot 1
MSG.PART3	bx1xxxxxx	b111 bxxxxxxxx1 xxx	Messages x4 card is plugged in to slot 3
MSG.PART4	bx1xxxxxx	b111 bxxxxxxx1x xxx	Messages x16 card plugged in to stack, Or x8 card plugged in to Lane 0/1, Or x4 card is plugged in to slot 0
MSG.PART5	bx1xxxxxx	b111 bxxxxxx1xx xxx	Messages x4 card is plugged in to slot 1
MSG.PART6	bx1xxxxxx	b111 bxxxxx1xxx xxx	Messages x8 card plugged in to Lane 2/3, Or x4 card is plugged in to slot 1
MSG.PART7	bx1xxxxxx	b111 bxxxx1xxxx xxx	Messages x4 card is plugged in to slot 3
CMPD.IOMMU0	b1xxxxxxx	b111 bxxx1xxxxx xxx	CmpD IOMMU - Type 0
CMPD.IOMMU1	b1xxxxxxx	b111 bxx1xxxxxx xxx	CmpD IOMMU - Type 1

Table 2-204. Unit Masks for TXN_REQ_OF_CPU (Sheet 5 of 5)

Extension	umask [15:8]	xtra [50:36]	Description
CMPD.PART0	b1xxxxxxx	b111 bxxxxxxxxx xx1	CmpD x16 card plugged in to stack, Or x8 card plugged in to Lane 0/1, Or x4 card is plugged in to slot 0
CMPD.PART1	b1xxxxxxx	b111 bxxxxxxxxx x1x	CmpD x4 card is plugged in to slot 1
CMPD.PART2	b1xxxxxxx	b111 bxxxxxxxxx 1xx	CmpD x8 card plugged in to Lane 2/3, Or x4 card is plugged in to slot 1
CMPD.PART3	b1xxxxxxx	b111 bxxxxxxxx1 xxx	CmpD x4 card is plugged in to slot 3
CMPD.PART4	b1xxxxxxx	b111 bxxxxxxx1x xxx	CmpD x16 card plugged in to stack, Or x8 card plugged in to Lane 0/1, Or x4 card is plugged in to slot 0
CMPD.PART5	b1xxxxxxx	b111 bxxxxxx1xx xxx	CmpD x4 card is plugged in to slot 1
CMPD.PART6	b1xxxxxxx	b111 bxxxxx1xxx xxx	CmpD x8 card plugged in to Lane 2/3, Or x4 card is plugged in to slot 1
CMPD.PART7	b1xxxxxxx	b111 bxxxx1xxxx xxx	CmpD x4 card is plugged in to slot 3

2.5 IIO Ring Port (IRP) Performance Monitoring

IIO Ring Port (IRP) is responsible for maintaining coherency for IIO traffic targeting coherent memory.

2.5.1 IRP Performance Monitoring Overview

Each IRP box supports event monitoring through two 48b wide counters (IRP{5-0}_MSR_PMON_CTR/CTL{1:0}). Each of these counters can be programmed to count any IRP event.

2.5.2 IRP Performance Monitoring Events

IRP provides events to track information related to all the traffic passing through its boundaries.

2.5.3 IRP Box Events Ordered By Code

The following table summarizes the directly measured IRP box events.

Table 2-205. Directly Measured IRP Box Events

Symbol Name	Event Code	Ctrs	Description
CACHE_TOTAL_OCCUPANCY	0xF	0-1	Total Write Cache Occupancy
CLOCKTICKS	0x1	0-1	IRP Clocks
FAF_FULL	0x17	0-1	FAF RF full
FAF_INSERTS	0x18	0-1	FAF - request insert from TC.
FAF_OCCUPANCY	0x19	0-1	FAF occupancy
FAF_TRANSACTIONS	0x16	0-1	FAF allocation sent to ADQ
IRP_ALL	0x20	0-1	
MISC0	0x1E	0-1	Misc Events - Set 0
MISC1	0x1F	0-1	Misc Events - Set 1
NOTHING	0x0	0-1	Enable
P2P_INSERTS	0x14	0-1	P2P Requests
P2P_OCCUPANCY	0x15	0-1	P2P Occupancy
P2P_TRANSACTIONS	0x13	0-1	P2P Transactions
TRANSACTIONS	0x11	0-1	Inbound Transaction Count
TxC_AK_INSERTS	0xB	0-1	AK Egress Allocations
TxC_BL_DRS_CYCLES_FULL	0x5	0-1	BL DRS Egress Cycles Full
TxC_BL_DRS_INSERTS	0x2	0-1	BL DRS Egress Inserts
TxC_BL_DRS_OCCUPANCY	0x8	0-1	BL DRS Egress Occupancy
TxC_BL_NCB_CYCLES_FULL	0x6	0-1	BL NCB Egress Cycles Full
TxC_BL_NCB_INSERTS	0x3	0-1	BL NCB Egress Inserts
TxC_BL_NCB_OCCUPANCY	0x9	0-1	BL NCB Egress Occupancy
TxC_BL_NCS_CYCLES_FULL	0x7	0-1	BL NCS Egress Cycles Full
TxC_BL_NCS_INSERTS	0x4	0-1	BL NCS Egress Inserts
TxC_BL_NCS_OCCUPANCY	0xA	0-1	BL NCS Egress Occupancy
TxR2_AD01_STALL_CREDIT_CYCLES	0x1C	0-1	
TxR2_AD0_STALL_CREDIT_CYCLES	0x1A	0-1	No AD0 Egress Credits Stalls
TxR2_AD1_STALL_CREDIT_CYCLES	0x1B	0-1	No AD1 Egress Credits Stalls
TxR2_BL_STALL_CREDIT_CYCLES	0x1D	0-1	No BL Egress Credit Stalls
TxS_DATA_INSERTS_NCB	0xD	0-1	Outbound Read Requests
TxS_DATA_INSERTS_NCS	0xE	0-1	Outbound Read Requests
TxS_REQUEST_OCCUPANCY	0xC	0-1	Outbound Request Queue Occupancy

2.5.4 IRP Box Performance Monitor Event List

The section enumerates 5th Gen Intel® Xeon® Scalable Processor performance monitoring events for the IRP box.

CACHE_TOTAL_OCCUPANCY

- **Title:**
- **Category:** WRITE_CACHE Events
- **Event Code:** 0xF
- **Register Restrictions :** 0-1
- **Definition:** Accumulates the number of reads and writes that are outstanding in the uncore in each cycle. This is effectively the sum of the READ_OCCUPANCY and WRITE_OCCUPANCY events.

Table 2-206. Unit Masks for CACHE_TOTAL_OCCUPANCY

Extension	umask [15:8]	Description
ANY	b00000001	Any Source Tracks all requests from any source port.
IV_Q	b00000010	Snoops
MEM	b00000100	Mem

CLOCKTICKS

- **Title:**
- **Category:** CLOCK Events
- **Event Code:** 0x1
- **Register Restrictions :** 0-1
- **Definition:**

FAF_FULL

- **Title:**
- **Category:** FAF Events
- **Event Code:** 0x17
- **Register Restrictions :** 0-1
- **Definition:** Inbound read transactions queue is full.

FAF_INSERTS

- **Title:**
- **Category:** FAF Events
- **Event Code:** 0x18
- **Register Restrictions :** 0-1
- **Definition:** Inbound read transactions.

FAF_OCCUPANCY

- **Title:**
- **Category:** FAF Events
- **Event Code:** 0x19
- **Register Restrictions :** 0-1
- **Definition:** Inbound read transaction queue occupancy.

FAF_TRANSACTIONS

- **Title:**
- **Category:** FAF Events
- **Event Code:** 0x16
- **Register Restrictions :** 0-1
- **Definition:** Inbound read transaction allocation.

IRP_ALL

- **Title:**
- **Category:** IRP Buffer Events
- **Event Code:** 0x20
- **Register Restrictions :** 0-1
- **Definition:** Counts all allocations/evicts from IRP to TXQ including dram reads, writes, p2p.

Table 2-207. Unit Masks for IRP_ALL

Extension	umask [15:8]	Description
INBOUND_INSERTS	b00000001	All Inserts Inbound (p2p + faf + cset)
EVICTS	b00000100	All Inserts Outbound (BL, AK, Snoops)

MISC0

- **Title:**
- **Category:** MISC Events
- **Event Code:** 0x1E
- **Register Restrictions :** 0-1
- **Definition:**

Table 2-208. Unit Masks for MISC0 (Sheet 1 of 2)

Extension	umask [15:8]	Description
FAST_REQ	b000000x1	Fastpath Requests Counts Fastpath Requests from ITC
FAST_REJ	b0000001x	Fastpath Rejects Counts Rejects of Fastpath Requests from ITC
2ND_RD_INSERT	bx00xx100	Cache Inserts of Read Transactions as Secondary
2ND_WR_INSERT	bx00x1x00	Cache Inserts of Write Transactions as Secondary Counts Conflicts (Subsequent Write to same address) for Write Requests from ITC
2ND_ATOMIC_INSERT	bx001xx00	Cache Inserts of Atomic Transactions as Secondary Counts Conflicts (Subsequent Atomic to same address) for Atomic Requests from ITC
FAST_XFER	bxx100000	Fastpath Transfers From Primary to Secondary Counts Fastpath Transfers From Primary to Secondary

Table 2-208. Unit Masks for MISC0 (Sheet 2 of 2)

Extension	umask [15:8]	Description
PF_ACK_HINT	bx1x00000	Prefetch Ack Hints From Primary to Secondary Counts Prefetch Ack Hints From Primary to Secondary
SLOWPATH_FWPF_NO_PR F	b1xx00000	Slow path FWPF did not find prefetch Counts Slow Path fetch from ITC which had a prefetch got rejected indicating a prefetch timeout

MISC1

- **Title:**
- **Category:** MISC Events
- **Event Code:** 0x1F
- **Register Restrictions :** 0-1
- **Definition:**

Table 2-209. Unit Masks for MISC1

Extension	umask [15:8]	Description
SLOW_I	b000xxxx1	Slow Transfer of I Line Snoop took cacheline ownership before write from data was committed.
SLOW_S	b000xxx1x	Slow Transfer of S Line Secondary received a transfer that did not have sufficient MESI state
SLOW_E	b000xx1xx	Slow Transfer of E Line Secondary received a transfer that did have sufficient MESI state
SLOW_M	b000x1xxx	Slow Transfer of M Line Snoop took cacheline ownership before write from data was committed.
LOST_FWD	b0001xxxx	Lost Forward Snoop pulled away ownership before a write was committed
SEC_RCVD_INVLD	bxx1x0000	Received Invalid Secondary received a transfer that did not have sufficient MESI state
SEC_RCVD_VLD	bx1xx0000	Received Valid Secondary received a transfer that did have sufficient MESI state

NOTHING

- **Title:**
- **Category:** CLOCK Events
- **Event Code:** 0x0
- **Register Restrictions :** 0-1
- **Definition:**

P2P_INSERTS

- **Title:**
- **Category:** P2P Events
- **Event Code:** 0x14
- **Register Restrictions :** 0-1

- **Definition:** P2P requests from the ITC

P2P_OCCUPANCY

- **Title:**
- **Category:** P2P Events
- **Event Code:** 0x15
- **Register Restrictions :** 0-1
- **Definition:** P2P B & S Queue Occupancy

P2P_TRANSACTIONS

- **Title:**
- **Category:** P2P Events
- **Event Code:** 0x13
- **Register Restrictions :** 0-1
- **Definition:**

Table 2-210. Unit Masks for P2P_TRANSACTIONS

Extension	umask [15:8]	Description
RD	bxxxxxxx1	P2P reads Counts Inbound P2P Reads
WR	bxxxxxx1x	P2P Writes Counts Inbound P2P Writes
MSG	bxxxxx1xx	P2P Message Counts Inbound P2P Message
CMPL	bxxxx1xxx	P2P completions Counts Inbound P2P Completions
REM	bxxx1xxxx	Match if remote only
REM_AND_TGT_MATCH	bxx1xxxxx	match if remote and target matches
LOC	bx1xxxxxx	match if local only
LOC_AND_TGT_MATCH	b1xxxxxxx	match if local and target matches

TRANSACTIONS

- **Title:**
- **Category:** TRANSACTIONS Events
- **Event Code:** 0x11
- **Register Restrictions :** 0-1
- **Definition:** Counts the number of “inbound” transactions from the IRP to the uncore. This can be filtered based on request type in addition to the source queue. Note the special filtering equation. We do OR-reduction on the request type.

Table 2-211. Unit Masks for TRANSACTIONS

Extension	umask [15:8]	Description
WRITES	bxxxxxx1x	Writes Tracks only write requests. Each write request should have a prefetch, so there is no need to explicitly track these requests. For writes that are tickled and have to retry, the counter will be incremented for each retry.
WR_PREF	bxxxx1xxx	Write Prefetches Tracks the number of write Prefetches.
ATOMIC	bxxx1xxxx	Atomic Tracks the number of atomic transactions
OTHER	bxx1xxxxx	Other Tracks the number of other kinds of transactions.
ORDERINGQ	bx1xxxxxx	Select Source Tracks only those requests that come from the port specified in the *IRP_PmonFilter.OrderingQ* register. This register allows one to select one specific queue. It is not possible to monitor multiple queues at a time. If this bit is not set, then requests from all sources will be counted.

TxC_AK_INSERTS

- **Title:**
- **Category:** AK Egress Events
- **Event Code:** 0xB
- **Register Restrictions :** 0-1
- **Definition:**

TxC_BL_DRS_CYCLES_FULL

- **Title:**
- **Category:** BL Egress Events
- **Event Code:** 0x5
- **Register Restrictions :** 0-1
- **Definition:** Counts number of cycles outbound DRS queue is full.

TxC_BL_DRS_INSERTS

- **Title:**
- **Category:** BL Egress Events
- **Event Code:** 0x2
- **Register Restrictions :** 0-1
- **Definition:** Counts outbound IDI data Responses (DRS) from core.

TxC_BL_DRS_OCCUPANCY

- **Title:**
- **Category:** BL Egress Events
- **Event Code:** 0x8
- **Register Restrictions :** 0-1
- **Definition:** Counts DRS queue occupancy.

TxC_BL_NCB_CYCLES_FULL

- **Title:**
- **Category:** BL Egress Events
- **Event Code:** 0x6
- **Register Restrictions :** 0-1
- **Definition:** Counts the number of cycles the outbound NCB queue is full.

TxC_BL_NCB_INSERTS

- **Title:**
- **Category:** BL Egress Events
- **Event Code:** 0x3
- **Register Restrictions :** 0-1
- **Definition:** Counts outbound Intel UPI NCB transactions from the core.

TxC_BL_NCB_OCCUPANCY

- **Title:**
- **Category:** BL Egress Events
- **Event Code:** 0x9
- **Register Restrictions :** 0-1
- **Definition:** Counts NCB queue occupancy.

TxC_BL_NCS_CYCLES_FULL

- **Title:**
- **Category:** BL Egress Events
- **Event Code:** 0x7
- **Register Restrictions :** 0-1
- **Definition:** Counts the number of cycles the outbound NCS queue is full.

TxC_BL_NCS_INSERTS

- **Title:**
- **Category:** BL Egress Events
- **Event Code:** 0x4
- **Register Restrictions :** 0-1
- **Definition:** Counts outbound UPI NCS transactions from the core.

TxC_BL_NCS_OCCUPANCY

- **Title:**
- **Category:** BL Egress Events
- **Event Code:** 0xA
- **Register Restrictions :** 0-1
- **Definition:** Counts NCS queue occupancy.

TxR2_AD01_STALL_CREDIT_CYCLES

- **Title:**
- **Category:** STALL_CYCLES Events
- **Event Code:** 0x1C
- **Register Restrictions :** 0-1
- **Definition:** Counts the number times when it is not possible to issue a request to the M2PCIe because there are no egress credits available on AD0, A1, or AD0 and AD1 both. Stalls on both AD0 and AD1 will count as 2.

TxR2_AD0_STALL_CREDIT_CYCLES

- **Title:**
- **Category:** STALL_CYCLES Events
- **Event Code:** 0x1A
- **Register Restrictions :** 0-1
- **Definition:** Counts the number times when it is not possible to issue a request to the M2PCIe because there are no AD0 egress credits available.

TxR2_AD1_STALL_CREDIT_CYCLES

- **Title:**
- **Category:** STALL_CYCLES Events
- **Event Code:** 0x1B
- **Register Restrictions :** 0-1
- **Definition:** Counts the number times when it is not possible to issue a request to the M2PCIe because there are no AD1 egress credits available.

TxR2_BL_STALL_CREDIT_CYCLES

- **Title:**
- **Category:** STALL_CYCLES Events
- **Event Code:** 0x1D
- **Register Restrictions :** 0-1
- **Definition:** Counts the number times when it is not possible to issue data to the R2PCIe because there are no BL egress credits available.

TxS_DATA_INSERTS_NCB

- **Title:**
- **Category:** OUTBOUND_REQUESTS Events
- **Event Code:** 0xD
- **Register Restrictions :** 0-1
- **Definition:** Counts the number of requests issued to the switch (towards the devices).

TxS_DATA_INSERTS_NCS

- **Title:**
- **Category:** OUTBOUND_REQUESTS Events
- **Event Code:** 0xE
- **Register Restrictions :** 0-1
- **Definition:** Counts the number of requests issued to the switch (towards the devices).

TxS_REQUEST_OCCUPANCY

- **Title:**
- **Category:** OUTBOUND_REQUESTS Events
- **Event Code:** 0xC
- **Register Restrictions :** 0-1
- **Definition:** Accumulates the number of outstanding outbound requests from the IRP to the switch (towards the devices). This can be used in conjunction with the allocations event in order to calculate average latency of outbound requests.

2.6 Intel® UPI Link Layer Performance Monitoring

The 5th Gen Intel® Xeon® Scalable Processor uses a coherent interconnect for scaling to multiple sockets known as Intel® Ultra Path Interconnect (UPI). Intel® UPI technology provides a cache coherent socket to socket external communication interface between processors. Intel® UPI is also used as a coherent communication interface between processors and OEM third party Node Controllers (XNC).

There are up to three Intel® UPI agents, each with its own mesh stop. These links can be connected to a single destination (such as in a DP) or multiple destinations. Therefore, it will be necessary to count Intel® UPI statistics for each agent separately.

The Intel® UPI module supports one Intel® UPI link (per mesh stop) and is comprised of the following layers for each Intel® UPI link:

- **Physical Layer** - The Intel® UPI Physical layer (PHY) is a hardware layer that lies between the link layer above it, and the physical wires that connect to other devices. The physical layer is further sub-divided into the logical and electrical sub-blocks.
- **Link Layer** - The Intel® UPI link layer bi-directionally converts between protocol layer messages and link layer flits, passes them through shared buffers, and manages the flow control information per virtual channel. The link layer also detects errors and retransmits packets on errors.
- **Routing Layer** - The Routing Layer is distributed among all agents that send Intel® UPI messages on the mesh (Intel® UPI, CHA, PCIe, IMC). The Intel® UPI module provides a routing function to determine the correct mesh stop from which to forward a given packet.
- **Protocol Layer** - The Intel® UPI module does not implement the protocol layer. A protocol agent is a proxy for some entity which injects, generates, or services Intel® UPI transactions such as memory requests, interrupts, and so forth. The protocol layer is implemented in the following modules: Coherency Home Agent (CHA), PCIe, Configuration Agent (Ubox). A Coherency Agent (CA) in the CHA both generates requests and services snoops. A Home Agent (HA) in the CHA services requests, generates snoops, and resolves conflicts. The CHA will sometimes behave as a CA, sometimes as an HA, and sometimes both at the same time. The PCIe module handles most IO proxy responsibilities. The Ubox handles internal configuration space and some other interrupt and messaging flows. An HA acts as proxy for DRAM, while the PCIe/Ubox handle all non-DRAM (NCB and NCS) requests.

The Intel UPI subsystem implements an Intel UPI port as a bi-directional interface. As such, the Intel UPI subsystem implements both transmit and receive interfaces and functionality. The Intel® UPI link layer is responsible for packetizing requests from the caching agent on the way out to the system interface. The Intel UPI link layer processes information at a flit granularity.

On the 5th Gen Intel® Xeon® Scalable Processor, Intel® UPI is split into two layers – M3UPI and the Intel UPI link layer. M3UPI (Section 2.9, "M3UPI Performance Monitoring") provides the interface to the Mesh for the link layer. M3UPI converts mesh packets (received from the CHA) into Intel UPI flits and vice-versa. M3UPI ingress is the point where remote Intel UPI VNA/VN0 link credits are acquired. The Intel UPI link layer passes flits through shared buffers, and manages their flow control. The link layer also detects errors, and on their occurrence, retransmits affected packets (corrected errors). Finally, the link layer delivers packets to the caching agent.

A single Intel UPI flit can pack up to three mesh packets in three slots. The Intel® UPI link layer has the ability to transmit up to three mesh packets per cycle in each direction. In order to accommodate this, many of the events in the link layer can increment by 0, 1, or 2 in each cycle. It is not possible to monitor Rx (received) and Tx (transmitted) flit information at the same time on the same counter.

Note: Flit slots are not symmetric in their ability to relay flit traffic. Any analysis of the Intel UPI bandwidth should keep this in mind.

2.6.1 Intel® UPI Performance Monitoring Overview

Each Intel® UPI link supports event monitoring through four 48b wide counters (U_Ly_PCI_PMON_CTR/CTL{3:0}). Each of these four counters can be programmed to count any Intel® UPI event. The Intel® UPI counters can increment by a maximum of 9b per cycle.

2.6.2 Additional Intel® UPI Performance Monitoring

2.6.2.1 Intel® UPI Extra Registers - Companions to PMON HW

The Intel® UPI box includes three registers that provide performance monitoring related information outside of the normal PMON infrastructure.

- A register that provides the current Intel® UPI transfer rate
- A 32b free running counter that counts the number of cycles that the Receive (Rx) side of the link is idle. Includes null cycles, and the cycles where the link is in L1 (that is, powered down).
- A 32b free running counter that counts the number of cycles that the Receive (Rx) side of the link is in LLR.

Table 2-212. UPI_RATE_STATUS Register – Field Definitions

Field	Bits	Attr	HW Reset Val	Description
rsv	31:3	RV	0	Reserved. SW must write to 0 else behavior is undefined.
UPI_rate	2:0	RO-V	11b	UPI Rate This reflects the current UPI rate setting into the PLL

Table 2-213. U_Ly_PCI_PMON_LINK_IDLE Register – Field Definitions

Field	Bits	Attr	HW Reset Val	Description
event_count	31:0	RW-V	0	32-bit performance event counter

Table 2-214. U_Ly_PCI_PMON_LINK_LLR Register – Field Definitions

Field	Bits	Attr	HW Reset Val	Description
event_count	31:0	RW-V	0	32-bit performance event counter

2.6.3 Intel® UPI LL Performance Monitoring Events

The Intel® UPI link layer provides events to gather information on topics such as:

- Tracking incoming (mesh bound)/outgoing (system bound) transactions.

2.6.4 Intel® UPI LL Box Events Ordered By Code

The following table summarizes the directly measured Intel® UPI LL Box events.

Table 2-215. Directly Measured Intel® UPI LL Box Events (Sheet 1 of 2)

Symbol Name	Event Code	Ctrs	Description
CLOCKTICKS	0x1	0-3	Number of KFCLKS
DIRECT_ATTEMPTS	0x12	0-3	Direct packet attempts
M3_BYP_BLOCKED	0x14	0-3	
M3_RXQ_BLOCKED	0x15	0-3	
M3_CRD_RETURN_BLOCKED	0x16	0-3	
FLOWQ_NO_VNA_CRD	0x18	0-3	
TxL_FLITS	0x2	0-3	Valid Flits Sent
PHY_INIT_CYCLES	0x20	0-3	Cycles where Phy is not in L0, L0c, L0p, L1
L1_POWER_CYCLES	0x21	0-3	Cycles in L1

Table 2-215. Directly Measured Intel® UPI LL Box Events (Sheet 2 of 2)

Symbol Name	Event Code	Ctrs	Description
POWER_L1_REQ	0x22	0-3	L1 Req (same as L1 Ack).
POWER_L1_NACK	0x23	0-3	L1 Req Nack
RxL0_POWER_CYCLES	0x24	0-3	Cycles in L0
RxL0P_POWER_CYCLES	0x25	0-3	Cycles in L0p
TxL0_POWER_CYCLES	0x26	0-3	Cycles in L0
TxL0P_POWER_CYCLES	0x27	0-3	Cycles in L0p
TxL0P_POWER_CYCLES_LL_ENTER	0x28	0-3	
TxL0P_POWER_CYCLES_M3_EXIT	0x29	0-3	
TxL0P_CLK_ACTIVE	0x2A	0-3	
RxL_FLITS	0x3	0-3	Valid Flits Received
RxL_INSERTS	0x30	0-3	RxQ Flit Buffer Allocations
RxL_BYPASSED	0x31	0-3	RxQ Flit Buffer Bypassed
RxL_OCCUPANCY	0x32	0-3	RxQ Occupancy - All Packets
RxL_SLOT_BYPASS	0x33	0-3	
RxL_CREDITS_CONSUMED_VNA	0x38	0-3	VNA Credit Consumed
RxL_CREDITS_CONSUMED_VN0	0x39	0-3	VN0 Credit Consumed
RxL_CREDITS_CONSUMED_VN1	0x3A	0-3	VN1 Credit Consumed
TxL_BASIC_HDR_MATCH	0x4	0-3	Matches on Transmit path of a UPI Port
TxL_INSERTS	0x40	0-3	Tx Flit Buffer Allocations
TxL_BYPASSED	0x41	0-3	Tx Flit Buffer Bypassed
TxL_OCCUPANCY	0x42	0-3	Tx Flit Buffer Occupancy
VNA_CREDIT_RETURN_OCCUPANCY	0x44	0-3	VNA Credits Pending Return - Occupancy
VNA_CREDIT_RETURN_BLOCKED_VN01	0x45	0-3	
REQ_SLOT2_FROM_M3	0x46	0-3	
RxL_BASIC_HDR_MATCH	0x5	0-3	Matches on Receive path of a UPI Port
RxL_CRC_LLR_REQ_TRANSMIT	0x8	0-3	LLR Requests Sent
RxL_CRC_ERRORS	0xB	0-3	CRC Errors Detected

2.6.5 Intel® UPI LL Box Common Metrics (Derived Events)

The following table summarizes metrics commonly calculated from Intel® UPI LL box events.

Table 2-216. Metrics Commonly Calculated From Intel® UPI LL Box Events

Symbol Name: Definition	Equation
DRS_E_FROM_UPI: DRS response in F or E states received from UPI in bytes. To calculate the total data response for each cache line state, itsnecessarytoaddthecontributionfromthreeflavors{DataC,DataC_FrcAckCnflt,DataC_Cmp}ofdataresponsepacketsforeachcachelinestate.'	RxL_BASIC_HDR_MATCH.{umask,opc}={0x1C,1} * 64
DRS_M_FROM_UPI: Data Response DataM packets received from UPI. Expressed in bytes	RxL_BASIC_HDR_MATCH.{umask,opc}={0x0C,1} * 64
DRS_WB_FROM_UPI: DRS writeback packets received from UPI in bytes. This is the sum of Wb{I,S,E} DRS packets	DRS_WbI_FROM_UPI + DRS_WbS_FROM_UPI + DRS_WbE_FROM_UPI
DRS_WbE_FROM_UPI: DRS writeback changeMtoEstate'packetsreceivedfromUPIinbytes'	RxL_BASIC_HDR_MATCH.{umask,opc}={0x2D,1} *64
DRS_WbI_FROM_UPI: DRS writeback changeMtoIstate'packetsreceivedfromUPIinbytes'	RxL_BASIC_HDR_MATCH.{umask,opc}={0x0D,1} *64
DRS_WbS_FROM_UPI: DRS writeback changeMtoSstate'packetsreceivedfromUPIinbytes'	RxL_BASIC_HDR_MATCH.{umask,opc}={0x1D,1} *64
NCB_DATA_FROM_UPI_TO_NODEx: NCB Data packets (Any - Interrupts) received from UPI sent to Node ID x'.Expressedinbytes'	RxL_BASIC_HDR_MATCH.{umask,endnid,dnid} = {0xE,1,x} * 64
PCT_LINK_CRC_RETRY_CYCLES: Percent of Cycles the UPI link layer is in retry mode due to CRC errors	RxL_CRC_CYCLES_IN_LLR / CLOCKTICKS
PCT_LINK_FULL_POWER_CYCLES: Percent of Cycles the UPI link is at Full Power	RxL0_POWER_CYCLES / CLOCKTICKS
PCT_LINK_HALF_DISABLED_CYCLES: Percent of Cycles the UPI link in power mode where half of the lanes are disabled.	RxL0P_POWER_CYCLES / CLOCKTICKS
PCT_LINK_SHUTDOWN_CYCLES: Percent of Cycles the UPI link is Shutdown	L1_POWER_CYCLES / CLOCKTICKS
UPI_SPEED: UPI Speed - In GT/s (GigaTransfers / Second) - Max UPI Bandwidth is 2 * ROUND (UPI Speed , 0)	ROUND ((CLOCKTICKS / TSC) * TSC_SPEED, 0) * (8 / 1000)

2.6.6 Intel® UPI LL Box Performance Monitor Event List

The section enumerates 5th Gen Intel® Xeon® Scalable Processor performance monitoring events for the Intel® UPI LL box.

CLOCKTICKS

- **Title:**
- **Category:** Clock Events
- **Event Code:** 0x1
- **Register Restrictions :** 0-3
- **Definition:** Counts the number of clocks in the Intel® UPI LL.

DIRECT_ATTEMPTS

- **Title:**
- **Category:** DIRECT2CORE Events
- **Event Code:** 0x12
- **Register Restrictions :** 0-3
- **Definition:** Counts the number of DRS packets that we attempted to do direct2core/direct2UPI on. There are four mutually exclusive filters. Filter [0] can be used to get successful spawns, while [1:3] provide the different failure cases. Note that this does not count packets that are not candidates for Direct2Core. The only candidates for Direct2Core are DRS packets destined for Cbos.

Table 2-217. Unit Masks for DIRECT_ATTEMPTS

Extension	umask [15:8]	Description
D2C	bxxxxxxx1	D2C
D2K	bxxxxxx1x	D2K

FLOWQ_NO_VNA_CRD

- **Title:**
- **Category:** LL to M3 Events
- **Event Code:** 0x18
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-218. Unit Masks for FLOWQ_NO_VNA_CRD

Extension	umask [15:8]	Description
AD_VNA_EQ0	bxxxxxxx1	
AD_VNA_EQ1	bxxxxxx1x	
AD_VNA_EQ2	bxxxxx1xx	
BL_VNA_EQ0	bxxxx1xxx	
AK_VNA_EQ0	bxxx1xxxx	
AK_VNA_EQ1	bxx1xxxxx	
AK_VNA_EQ2	bx1xxxxxx	
AK_VNA_EQ3	b1xxxxxxx	

L1_POWER_CYCLES

- **Title:**
- **Category:** Power Events
- **Event Code:** 0x21
- **Register Restrictions :** 0-3
- **Definition:** Number of Intel® UPI QFCLK cycles spent in L1 power mode. L1 is a mode that totally shuts down an Intel® UPI link. Use edge detect to count the number of instances when the Intel® UPI link entered L1. Link power states are per link and per direction, so for example the Tx direction could be in one state while Rx was in another. Because L1 totally shuts down the link, it takes a good amount of time to exit this mode.

M3_BYP_BLOCKED

- **Title:**
- **Category:** LL to M3 Events
- **Event Code:** 0x14
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-219. Unit Masks for M3_BYP_BLOCKED

Extension	umask [15:8]	Description
FLOWQ_AD_VNA_LE2	bxxxxxxx1	
FLOWQ_BL_VNA_EQ0	bxxxxxx1x	
FLOWQ_AK_VNA_LE3	bxxxxx1xx	
BGF_CRD	bxxxx1xxx	
GV_BLOCK	bxxx1xxxx	

M3_CRD_RETURN_BLOCKED

- **Title:**
- **Category:** LL to M3 Events
- **Event Code:** 0x16
- **Register Restrictions :** 0-3
- **Definition:**

M3_RXQ_BLOCKED

- **Title:**
- **Category:** LL to M3 Events
- **Event Code:** 0x15
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-220. Unit Masks for M3_RXQ_BLOCKED

Extension	umask [15:8]	Description
FLOWQ_AD_VNA_LE2	bxxxxxxx1	
FLOWQ_AD_VNA_BTW_2_THRESH	bxxxxxx1x	
FLOWQ_BL_VNA_EQ0	bxxxxx1xx	
FLOWQ_BL_VNA_BTW_0_THRESH	bxxxx1xxx	
FLOWQ_AK_VNA_LE3	bxxx1xxxx	
BGF_CRD	bxx1xxxxx	
GV_BLOCK	bx1xxxxxx	

PHY_INIT_CYCLES

- **Title:**
- **Category:** Power Events
- **Event Code:** 0x20
- **Register Restrictions :** 0-3
- **Definition:**

POWER_L1_NACK

- **Title:**
- **Category:** Power Events
- **Event Code:** 0x23
- **Register Restrictions :** 0-3
- **Definition:** Counts the number of times a link sends/receives a LinkReqNAck. When the Intel® UPI links would like to change power state, the Tx side initiates a request to the Rx side requesting to change states. This requests can either be accepted or denied. If the Rx side replies with an ACK, the power mode will change. If it replies with NACK, no change will take place. This can be filtered based on Rx and Tx. An Rx LinkReqNAck refers to receiving an NACK (meaning this agents Tx originally requested the power change). A TxLinkReq NACK refers to sending this command (meaning the peer agent’s Tx originally requested the power change and this agent accepted it).

POWER_L1_REQ

- **Title:**
- **Category:** Power Events
- **Event Code:** 0x22
- **Register Restrictions :** 0-3
- **Definition:** Counts the number of times a link sends/receives a LinkReqAck. When the Intel® UPI links would like to change power state, the Tx side initiates a request to the Rx side requesting to change states. This requests can either be accepted or denied. If the Rx side replies with an ACK, the power mode will change. If it replies with NACK, no change will take place. This can be filtered based on Rx and Tx. A Rx LinkReqAck refers to receiving an ACK (meaning this agent’s Tx originally requested

the power change). A Tx LinkReqAck refers to sending this command (meaning the peer agent's Tx originally requested the power change and this agent accepted it).

REQ_SLOT2_FROM_M3

- **Title:**
- **Category:** VNA Credit Events
- **Event Code:** 0x46
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-221. Unit Masks for REQ_SLOT2_FROM_M3

Extension	umask [15:8]	Description
VNA	bxxxxxxx1	
VN0	bxxxxxx1x	
VN1	bxxxxx1xx	
ACK	bxxxx1xxx	

RxL0P_POWER_CYCLES

- **Title:**
- **Category:** Rx Power Events
- **Event Code:** 0x25
- **Register Restrictions :** 0-3
- **Definition:** Number of Intel® UPI QFCLK cycles spent in L0p power mode. L0p is a mode where we disable half of the Intel® UPI lanes, decreasing our bandwidth in order to save power. It increases snoop and data transfer latencies and decreases overall bandwidth. This mode can be very useful in NUMA optimized workloads that largely only utilize Intel® UPI for snoops and their responses. Use edge detect to count the number of instances when the Intel® UPI link entered L0p. Link power states are per link and per direction, so for example the Tx direction could be in one state while Rx was in another.

RxL0_POWER_CYCLES

- **Title:**
- **Category:** Rx Power Events
- **Event Code:** 0x24
- **Register Restrictions :** 0-3
- **Definition:** Number of Intel® UPI QFCLK cycles spent in L0 power mode in the Link Layer. L0 is the default mode which provides the highest performance with the most power. Use edge detect to count the number of instances that the link entered L0. Link power states are per link and per direction, so for example the Tx direction could be in one state while Rx was in another. The Phy layer sometimes leaves L0 for training, which will not be captured by this event.

RxL_BASIC_HDR_MATCH

- **Title:**
- **Category:** FLIT match Events
- **Event Code:** 0x5
- **Register Restrictions :** 0-3
- **Definition:** Matches on the receive path of a Intel® UPI port.

Table 2-222. Unit Masks for RxL_BASIC_HDR_MATCH

Extension	umask [15:8]	xtra [57:32]	Description
REQ	bxxxx1000	0x0	Request
REQ_OPC	bxxxx1000	0x1	Request, Match Opcode
SNP	bxxxx1001	0x0	Snoop
SNP_OPC	bxxxx1001	0x1	Snoop, Match Opcode
RSP_NODATA	bxxxx1010	0x0	Response - No Data
RSP_NODATA_OPC	bxxxx1010	0x1	Response - No Data, Match Opcode
RSP_DATA	bxxxx1100	0x0	Response - Data
RSP_DATA_OPC	bxxxx1100	0x1	Response - Data, Match Opcode
WB	bxxxx1101	0x0	Write back
WB_OPC	bxxxx1101	0x1	Write back, Match Opcode
NCB	bxxxx1110	0x0	Non-Coherent Bypass
NCB_OPC	bxxxx1110	0x1	Non-Coherent Bypass, Match Opcode
NCS	bxxxx1111	0x0	Non-Coherent Standard
NCS_OPC	bxxxx1111	0x1	Non-Coherent Standard, Match Opcode
RSPI	b00101010	0x1	Response - Invalid
RSPCNFLT	b10101010	0x1	Response - Conflict

RxL_BYPASSED

- **Title:**
- **Category:** RXQ Events
- **Event Code:** 0x31
- **Register Restrictions :** 0-3
- **Definition:** Counts the number of times that an incoming flit was able to bypass the flit buffer and pass directly across the BGF and into the egress. This is a latency optimization, and should generally be the common case. If this value is less than the

number of flits transfered, it implies that there was queuing getting onto the ring, and thus the transactions saw higher latency.

Table 2-223. Unit Masks for RxL_BYPASSED

Extension	umask [15:8]	Description
SLOT0	bxxxxxxx1	Slot 0
SLOT1	bxxxxxx1x	Slot 1
SLOT2	bxxxxx1xx	Slot 2

RxL_CRC_ERRORS

- **Title:**
- **Category:** CRC_ERRORS_RX Events
- **Event Code:** 0xB
- **Register Restrictions :** 0-3
- **Definition:** Number of CRC errors detected in the Intel® UPI Agent. Each Intel® UPI flit incorporates 8 bits of CRC for error detection. This counts the number of flits where the CRC was able to detect an error. After an error has been detected, the Intel® UPI agent will send a request to the transmitting socket to resend the flit (as well as any flits that came after it).

RxL_CRC_LLR_REQ_TRANSMIT

- **Title:**
- **Category:** CRC_ERRORS_RX Events
- **Event Code:** 0x8
- **Register Restrictions :** 0-3
- **Definition:** Number of LLR requests were transmitted. This should generally be ≤ the number of CRC errors detected. If multiple errors are detected before the Rx side receives a LLC_REQ_ACK from the Tx side, there is no need to send more LLR_REQ_NACKs.

RxL_CREDITS_CONSUMED_VN0

- **Title:**
- **Category:** RX_CREDITS_CONSUMED Events
- **Event Code:** 0x39
- **Register Restrictions :** 0-3
- **Definition:** Counts the number of times that an RxQ VN0 credit was consumed (that is, message uses a VN0 credit for the Rx buffer). This includes packets that went through the RxQ and those that were bypassed.

RxL_CREDITS_CONSUMED_VN1

- **Title:**
- **Category:** RX_CREDITS_CONSUMED Events
- **Event Code:** 0x3A
- **Register Restrictions :** 0-3
- **Definition:** Counts the number of times that an RxQ VN1 credit was consumed (that is, message uses a VN1 credit for the Rx buffer). This includes packets that went through the RxQ and those that were bypassed.

RxL_CREDITS_CONSUMED_VNA

- **Title:**
- **Category:** RX_CREDITS_CONSUMED Events
- **Event Code:** 0x38
- **Register Restrictions :** 0-3
- **Definition:** Counts the number of times that an RxQ VNA credit was consumed (that is, message uses a VNA credit for the Rx buffer). This includes packets that went through the RxQ and those that were bypassed.

RxL_FLITS

- **Title:**
- **Category:** Flit Events
- **Event Code:** 0x3
- **Register Restrictions :** 0-3
- **Definition:** Shows legal flit time (hides impact of L0p and L0c).

Table 2-224. Unit Masks for RxL_FLITS

Extension	umask [15:8]	Description
SLOT0	bxxxxxxx1	Slot 0 Count Slot 0 - Other mask bits determine types of headers to count.
SLOT1	bxxxxxx1x	Slot 1 Count Slot 1 - Other mask bits determine types of headers to count.
SLOT2	bxxxxx1xx	Slot 2 Count Slot 2 - Other mask bits determine types of headers to count.
DATA	bxxxx1xxx	Data Count Data Flits (which consume all slots), but how much to count is based on Slot0-2 mask, so count can be 0-3 depending on which slots are enabled for counting.
ALL_DATA	b00001111	All Data
LLCRD	bxxx1xxxx	LLCRD Not Empty Enables counting of LLCRD (with non-zero payload). This only applies to slot 2 since LLCRD is only allowed in slot 2
NULL	bxx1xxxxx	Slot NULL or LLCRD Empty LLCRD with all zeros is treated as NULL. Slot 1 is not treated as NULL if slot 0 is a dual slot. This can apply to slot 0,1, or 2.
ALL_NULL	b00100111	Null flits received from any slot
LLCTRL	bx1xxxxxx	LLCTRL Equivalent to an idle packet. Enables counting of slot 0 LLCTRL messages.
IDLE	b01000111	Idle
PROTHDR	b1xxxxxxx	Protocol Header Enables count of protocol headers in slot 0,1,2 (depending on slot uMask bits)
NON_DATA	b10010111	All Non Data

RxL_INSERTS

- **Title:**
- **Category:** RXQ Events
- **Event Code:** 0x30
- **Register Restrictions :** 0-3
- **Definition:** Number of allocations into the Intel UPI Rx flit buffer. Generally, when data is transmitted across Intel UPI, it will bypass the RxQ and pass directly to the ring interface. If things back up getting transmitted onto the ring, however, it may need to allocate into this buffer, thus increasing the latency. This event can be used in conjunction with the *Flit Buffer Occupancy* event in order to calculate the average flit buffer lifetime.

Table 2-225. Unit Masks for RxL_INSERTS

Extension	umask [15:8]	Description
SLOT0	bxxxxxxx1	Slot 0
SLOT1	bxxxxxx1x	Slot 1
SLOT2	bxxxxx1xx	Slot 2

RxL_OCCUPANCY

- **Title:**
- **Category:** RXQ Events
- **Event Code:** 0x32
- **Register Restrictions :** 0-3
- **Definition:** Accumulates the number of elements in the Intel UPI RxQ in each cycle. Generally, when data is transmitted across Intel UPI, it will bypass the RxQ and pass directly to the ring interface. If things back up getting transmitted onto the ring, however, it may need to allocate into this buffer, thus increasing the latency. This event can be used in conjunction with the *Flit Buffer Not Empty* event to calculate average occupancy, or with the *Flit Buffer Allocations* event to track average lifetime.

Table 2-226. Unit Masks for RxL_OCCUPANCY

Extension	umask [15:8]	Description
SLOT0	bxxxxxxx1	Slot 0
SLOT1	bxxxxxx1x	Slot 1
SLOT2	bxxxxx1xx	Slot 2

RxL_SLOT_BYPASS

- **Title:**
- **Category:** RxL Credit Events
- **Event Code:** 0x33
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-227. Unit Masks for RxL_SLOT_BYPASS

Extension	umask [15:8]	Description
S0_RXQ1	bxxxxxxx1	
S0_RXQ2	bxxxxxx1x	
S1_RXQ0	bxxxxx1xx	
S1_RXQ2	bxxxx1xxx	
S2_RXQ0	bxxx1xxxx	
S2_RXQ1	bxx1xxxxx	

TxL0P_CLK_ACTIVE

- **Title:**
- **Category:** Tx Power Events
- **Event Code:** 0x2A
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-228. Unit Masks for TxL0P_CLK_ACTIVE

Extension	umask [15:8]	Description
CFG_CTL	bxxxxxxx1	
RXQ	bxxxxxx1x	
RXQ_BYPASS	bxxxxx1xx	
RXQ_CRED	bxxxx1xxx	
TXQ	bxxx1xxxx	
RETRY	bxx1xxxxx	
DFX	bx1xxxxxx	
SPARE	b1xxxxxxx	

TxL0P_POWER_CYCLES

- **Title:**
- **Category:** Tx Power Events
- **Event Code:** 0x27
- **Register Restrictions :** 0-3

- **Definition:** Number of Intel UPI QFCLK cycles spent in L0p power mode. L0p is a mode where we disable half of the Intel UPI lanes, decreasing our bandwidth in order to save power. It increases snoop and data transfer latencies and decreases overall bandwidth. This mode can be very useful in NUMA optimized workloads that largely only utilize Intel UPI for snoops and their responses. Use edge detect to count the number of instances when the Intel UPI link entered L0p. Link power states are per link and per direction, so for example the Tx direction could be in one state while Rx was in another.

TxL0P_POWER_CYCLES_LL_ENTER

- **Title:**
- **Category:** Tx Power Events
- **Event Code:** 0x28
- **Register Restrictions :** 0-3
- **Definition:**

TxL0P_POWER_CYCLES_M3_EXIT

- **Title:**
- **Category:** Tx Power Events
- **Event Code:** 0x29
- **Register Restrictions :** 0-3
- **Definition:**

TxL0_POWER_CYCLES

- **Title:**
- **Category:** Tx Power Events
- **Event Code:** 0x26
- **Register Restrictions :** 0-3
- **Definition:** Number of Intel UPI QFCLK cycles spent in L0 power mode in the link layer. L0 is the default mode which provides the highest performance with the most power. Use edge detect to count the number of instances that the link entered L0. Link power states are per link and per direction, so for example the Tx direction could be in one state while Rx was in another. The Phy layer sometimes leaves L0 for training, which will not be captured by this event.

TxL_BASIC_HDR_MATCH

- **Title:**
- **Category:** FLIT match Events
- **Event Code:** 0x4
- **Register Restrictions :** 0-3
- **Definition:** Matches on the transmit path of an Intel UPI port.

Table 2-229. Unit Masks for TxL_BASIC_HDR_MATCH (Sheet 1 of 2)

Extension	umask [15:8]	xtra [57:32]	Description
REQ	bxxxx1000	0x0	Request
REQ_OPC	bxxxx1000	0x1	Request, Match Opcode
SNP	bxxxx1001	0x0	Snoop

Table 2-229. Unit Masks for TxL_BASIC_HDR_MATCH (Sheet 2 of 2)

Extension	umask [15:8]	xtra [57:32]	Description
SNP_OPC	bxxxx1001	0x1	Snoop, Match Opcode
RSP_NODATA	bxxxx1010	0x0	Response - No Data
RSP_NODATA_OPC	bxxxx1010	0x1	Response - No Data, Match Opcode
RSP_DATA	bxxxx1100	0x0	Response - Data
RSP_DATA_OPC	bxxxx1100	0x1	Response - Data, Match Opcode
WB	bxxxx1101	0x0	Write back
WB_OPC	bxxxx1101	0x1	Write back, Match Opcode
NCB	bxxxx1110	0x0	Non-Coherent Bypass
NCB_OPC	bxxxx1110	0x1	Non-Coherent Bypass, Match Opcode
NCS	bxxxx1111	0x0	Non-Coherent Standard
NCS_OPC	bxxxx1111	0x1	Non-Coherent Standard, Match Opcode
RSPI	b00101010	0x1	Response - Invalid
RSPCNFLT	b10101010	0x1	Response - Conflict

TxL_BYPASSED

- **Title:**
- **Category:** TxL Events
- **Event Code:** 0x41
- **Register Restrictions :** 0-3
- **Definition:** Counts the number of times that an incoming flit was able to bypass the Tx flit buffer and pass directly out the Intel UPI Link. Generally, when data is transmitted across Intel UPI, it will bypass the TxQ and pass directly to the link. However, the TxQ will be used with L0p and when an LLR occurs, increasing latency to transfer out to the link.

TxL_FLITS

- **Title:**
- **Category:** Flit Events
- **Event Code:** 0x2
- **Register Restrictions :** 0-3
- **Definition:** Shows legal flit time (hides impact of L0p and L0c).

Table 2-230. Unit Masks for TxL_FLITS

Extension	umask [15:8]	Description
SLOT0	bxxxxxxx1	Slot 0 Count Slot 0 - Other mask bits determine types of headers to count.
SLOT1	bxxxxxx1x	Slot 1 Count Slot 1 - Other mask bits determine types of headers to count.
SLOT2	bxxxxx1xx	Slot 2 Count Slot 2 - Other mask bits determine types of headers to count.
DATA	bxxxx1xxx	Data Count Data Flits (which consume all slots), but how much to count is based on Slot0-2 mask, so count can be 0-3 depending on which slots are enabled for counting.
ALL_DATA	b00001111	All Data
LLCRD	bxxx1xxxx	LLCRD Not Empty Enables counting of LLCRD (with non-zero payload). This only applies to slot 2 since LLCRD is only allowed in slot 2
NULL	bxx1xxxxx	Slot NULL or LLCRD Empty LLCRD with all zeros is treated as NULL. Slot 1 is not treated as NULL if slot 0 is a dual slot. This can apply to slot 0,1, or 2.
ALL_NULL	b00100111	Idle
LLCTRL	bx1xxxxxx	LLCTRL Equivalent to an idle packet. Enables counting of slot 0 LLCTRL messages.
IDLE	b01000111	
PROTHDR	b1xxxxxxx	Protocol Header Enables count of protocol headers in slot 0,1,2 (depending on slot umask bits)
NON_DATA	b10010111	Null flits transmitted to any slot

TxL_INSERTS

- **Title:**
- **Category:** TxL Events
- **Event Code:** 0x40
- **Register Restrictions :** 0-3
- **Definition:** Number of allocations into the Intel UPI Tx flit buffer. Generally, when data is transmitted across Intel UPI, it will bypass the TxQ and pass directly to the link. However, the TxQ will be used with L0p and when LLR occurs, increasing latency to transfer out to the link. This event can be used in conjunction with the *Flit Buffer Occupancy* event in order to calculate the average flit buffer lifetime.

TxL_OCCUPANCY

- **Title:**
- **Category:** TxL Events
- **Event Code:** 0x42
- **Register Restrictions :** 0-3
- **Definition:** Accumulates the number of flits in the TxQ. Generally, when data is transmitted across Intel UPI, it will bypass the TxQ and pass directly to the link. However, the TxQ will be used with L0p and when LLR occurs, increasing latency to transfer out to the link. This can be used with the cycles not empty event to track average occupancy, or the allocations event to track average lifetime in the TxQ.

VNA_CREDIT_RETURN_BLOCKED_VN01

- **Title:**
- **Category:** VNA Credit Events
- **Event Code:** 0x45
- **Register Restrictions :** 0-3
- **Definition:**

VNA_CREDIT_RETURN_OCCUPANCY

- **Title:**
- **Category:** VNA Credit Events
- **Event Code:** 0x44
- **Register Restrictions :** 0-3
- **Definition:** Number of VNA credits in the Rx side that are waiting to be returned back across the link.

2.7 M2M Performance Monitoring

M2M blocks manage the interface between the mesh (operating on both mesh and the SMI3 protocol) and the memory controllers. M2M acts as intermediary between the local CHA issuing memory transactions to its attached memory controller. Commands from M2M to the MC are serialized by a scheduler and only one can cross the interface at a time.

2.7.1 M2M Performance Monitoring Overview

Each M2M box supports event monitoring through four 48b wide counters (M2Mn_PCI_PMON_CTR/CTL{3:0}). Each of these four counters can be programmed to count almost any M2M event. M2M PMON also includes mask or match registers that allow the user to match packets of traffic heading to DRAM or heading to the mesh, according to various standard packet fields such as message class, opcode, and so forth.

2.7.2 Additional M2M Performance Monitoring

Table 2-231. Additional M2M Performance Monitoring Registers (PCICFG) (Sheet 1 of 2)

Register Name	PCICFG Address	Size (bits)	Description
PCICFG Base Address	Dev:Func DeviceID		
M2M 0 PMON Registers	D12:F0 0x324A		
M2M 1 PMON Registers	D13:F0 0x324A		
M2M 2 PMON Registers	D14:F0 0x324A		
M2M 3 PMON Registers	D15:F0 0x324A		

Table 2-231. Additional M2M Performance Monitoring Registers (PCICFG) (Sheet 2 of 2)

Register Name	PCICFG Address	Size (bits)	Description
Box-Level Filters			
M2Mn_PCI_PMON_OPCODE_MM	4A0	32	
M2Mn_PCI_PMON_ADDRMASK0	498	32	
M2Mn_PCI_PMON_ADDRMASK1	49C	32	
M2Mn_PCI_PMON_ADDRMATCH0	490	32	
M2Mn_PCI_PMON_ADDRMATCH1	494	32	

2.7.2.1 M2M Filter Registers

In addition to generic event counting, each M2M has several registers that allow a user to opcode match and address match packet traffic into and out of M2M. The filter registers report the number of matches through the PKT_MATCH event. The opcode mask and match register provides the ability to simultaneously filter incoming and outgoing traffic.

Figure 2-7. M2M PMON Opcode Filter Register

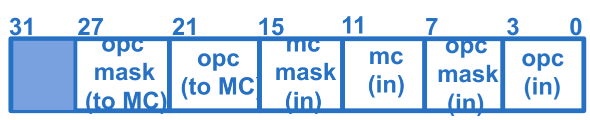


Table 2-232. M2Mn_PCI_PMON_OPCODE_MM Register – Field Definitions

Field	Bits	Atrtr	HW Reset Val	Description
rsv	31:28	RV	0	Reserved. SW must set to 0 else behavior is undefined
mcopc_mask	27:22	RW	0	Outgoing Opcode Mask encoding - See the SMI3 encodings in the Packet Match table
mcopc	21:16	RW	0	Outgoing Opcode Match encoding - See the SMI3 encodings in the Packet Match table
mc_mask	15:12	RW	0	Incoming Message Class Mask
mc	11:8	RW	0	Incoming Message Class Match 4'b0000: REQ (AD) 4'b0010: RSP (AD) 4'b1010: NCB (BL) 4'b1011: NCS (BL) 4'b1100: WB (BL)
opc_mask	7:4	RW	0	Incoming Opcode Mask encoding - See the SMI3 encodings in the Packet Match table
opc	3:0	RW	0	Incoming Opcode Match encoding - See the SMI3 encodings in the Packet Match table

Table 2-233. M2Mn_PCI_PMON_ADDR{MASK,MATCH}0 Register – Field Definitions

Field	Bits	Atrtr	HW Reset Val	Description
addr	31:0	RW	0	LSB [31:0] of transaction address to mask/match on

Table 2-234. M2Mn_PCI_PMON_ADDR{MASK,MATCH}1 Register – Field Definitions

Field	Bits	Atrtr	HW Reset Val	Description
rsv	31:15	RV	0	Reserved SW must set to 0 else behavior is undefined
addr	14:0	RW	0	MSB [45:32] of transaction address to mask/match on

2.7.3 M2M Performance Monitoring Events

M2M provides events to track information related to all the traffic passing through its boundaries.

- Mesh Stop Events.
  To track ingress/egress traffic and mesh utilization (broken down by direction and ring type) statistics. See Section 2.1, "Mesh Performance Monitoring".
- IMC credit tracking - credits rejected, acquired and used all broken down by message class.
- Reads and Writes issued to the IMC.
- D2C and D2U Events.
  To distinguish traffic thats able to use the available shortcuts from traffic that cannot.
- Directory Events
  To distinguish traffic thats able to use the available shortcuts from traffic that cannot.
- *Tracker* and *Scoreboard* events.

2.7.4 M2M Box Events Ordered By Code

The following table summarizes the directly measured M2M Box events.

Table 2-235. Directly Measured M2M Box Events (Sheet 1 of 4)

Symbol Name	Event Code	Ctrs	Description
AG0_AD_CRD_ACQUIRED0	0x80		CMS Agent0 AD Credits Acquired
AG0_AD_CRD_ACQUIRED1	0x81		CMS Agent0 AD Credits Acquired
AG0_AD_CRD_OCCUPANCY0	0x82		CMS Agent0 AD Credits Occupancy
AG0_AD_CRD_OCCUPANCY1	0x83		CMS Agent0 AD Credits Occupancy
AG0_BL_CRD_ACQUIRED0	0x88		CMS Agent0 BL Credits Acquired
AG0_BL_CRD_ACQUIRED1	0x89		CMS Agent0 BL Credits Acquired

Table 2-235. Directly Measured M2M Box Events (Sheet 2 of 4)

Symbol Name	Event Code	Ctrs	Description
AG0_BL_CRD_OCCUPANCY0	0x8A		CMS Agent0 BL Credits Occupancy
AG0_BL_CRD_OCCUPANCY1	0x8B		CMS Agent0 BL Credits Occupancy
AG1_AD_CRD_ACQUIRED0	0x84		CMS Agent1 AD Credits Acquired
AG1_AD_CRD_ACQUIRED1	0x85		CMS Agent1 AD Credits Acquired
AG1_AD_CRD_OCCUPANCY0	0x86		CMS Agent1 AD Credits Occupancy
AG1_AD_CRD_OCCUPANCY1	0x87		CMS Agent1 AD Credits Occupancy
AG1_BL_CRD_ACQUIRED0	0x8C		CMS Agent1 BL Credits Acquired
AG1_BL_CRD_ACQUIRED1	0x8D		CMS Agent1 BL Credits Acquired
AG1_BL_CRD_OCCUPANCY0	0x8E		CMS Agent1 BL Credits Occupancy
AG1_BL_CRD_OCCUPANCY1	0x8F		CMS Agent1 BL Credits Occupancy
BYPASS_M2M_EGRESS	0x15	0-3	M2M to iMC Bypass
CLOCKTICKS	0x1	0-3	Clock ticks of the mesh to memory (M2M)
CMS_CLOCKTICKS	0xC0		CMS Clock ticks
DIRECT2CORE_NOT_TAKEN_DIRSTATE	0x17	0-3	Cycles when direct to core mode, which bypasses the CHA, was disabled
DIRECT2CORE_NOT_TAKEN_NOTFORKED	0x4A	0-3	Counts the time when FM did not do d2c for fill reads (cross tile case)
DIRECT2CORE_TXN_OVERRIDE	0x18	0-3	Number of reads in which direct to core transaction was overridden
DIRECT2UPI_NOT_TAKEN_CREDITS	0x1B	0-3	Direct to UPI Transactions - Ignored due to lack of credits
DIRECT2UPI_NOT_TAKEN_DIRSTATE	0x1A	0-3	Cycles when Direct2UPI was Disabled
DIRECT2UPI_TXN_OVERRIDE	0x1C	0-3	Number of times a direct to UPI transaction was overridden
DIRECTORY_HIT	0x1D	0-3	Directory Hit
DIRECTORY_LOOKUP	0x20	0-3	Multi-socket cache line Directory Lookups
DIRECTORY_MISS	0x1E	0-3	Directory Miss
DIRECTORY_UPDATE	0x21	0-3	Multi-socket cache line Directory Updates
EGRESS_ORDERING	0xBA		Egress Blocking due to Ordering requirements
HORZ_RING_AD_IN_USE	0xB6		Horizontal AD Ring In Use
HORZ_RING_AKC_IN_USE	0xBB		Horizontal AK Ring In Use
TxC_BL	0x0E		BL Egress (to CMS)
TxC_AD_CREDITS	0x09		AD Egress Credits Occupancy
TxC_AD_CREDITS	0x08		AD Egress Credits
HORZ_RING_AK_IN_USE	0xB7		Horizontal AK Ring In Use
HORZ_RING_BL_IN_USE	0xB8		Horizontal BL Ring in Use
HORZ_RING_IV_IN_USE	0xB9		Horizontal IV Ring in Use
IMC_READS	0x24	0-3	
IMC_WRITES	0x25	0-3	
MISC_EXTERNAL	0xE6		Miscellaneous Events (mostly from MS2IDI)
Nothing	0x0		Counts no events
PREFCAM_DEALLOCS	0x57	0-3	Prefetch CAM Deallocs

Table 2-235. Directly Measured M2M Box Events (Sheet 3 of 4)

Symbol Name	Event Code	Ctrs	Description
PREFCAM_DEMAND_DROPS	0x58	0-3	Data Prefetches Dropped
PREFCAM_DROP_REASONS_CH0	0x59	0-3	Data Prefetches Dropped Ch0 - Reasons
PREFCAM_DROP_REASONS_CH1	0x5A	0-3	Data Prefetches Dropped Ch1 - Reasons
PREFCAM_INSERTS	0x56	0-3	Prefetch CAM Inserts
PREFCAM_RxC_DEALLOCS	0x62	0-3	
RING_BOUNCES_HORZ	0xAC		Messages that bounced on the Horizontal Ring.
DISTRESS_ASSERTED	0xAF		Distress signal asserted
RING_BOUNCES_VERT	0xAA		Messages that bounced on the Vertical Ring.
RING_SINK_STARVED_HORZ	0xAD		Sink Starvation on Horizontal Ring
RING_SINK_STARVED_VERT	0xAB		Sink Starvation on Vertical Ring
RING_SRC_THRTL	0xAE		Source Throttle
RxR_BUSY_STARVED	0xE5		Transgress Injection Starvation
RxR_BYPASS	0xE2		Transgress Ingress Bypass
RxR_CRD_STARVED	0xE3		Transgress Injection Starvation
RxR_CRD_STARVED_1	0xE4		Transgress Injection Starvation
RxR_INSERTS	0xE1		Transgress Ingress Allocations
RxR_OCCUPANCY	0xE0		Transgress Ingress Occupancy
STALL0_NO_TxR_HORZ_CRD_AD_AG0	0xD0		Stall on No AD Agent0 Transgress Credits
STALL0_NO_TxR_HORZ_CRD_AD_AG1	0xD2		Stall on No AD Agent1 Transgress Credits
STALL0_NO_TxR_HORZ_CRD_BL_AG0	0xD4		Stall on No BL Agent0 Transgress Credits
STALL0_NO_TxR_HORZ_CRD_BL_AG1	0xD6		Stall on No BL Agent1 Transgress Credits
STALL1_NO_TxR_HORZ_CRD_AD_AG0	0xD1		Stall on No AD Agent0 Transgress Credits
STALL1_NO_TxR_HORZ_CRD_AD_AG1	0xD3		Stall on No AD Agent1 Transgress Credits
STALL1_NO_TxR_HORZ_CRD_BL_AG0	0xD5		Stall on No BL Agent0 Transgress Credits
STALL1_NO_TxR_HORZ_CRD_BL_AG1	0xD7		Stall on No BL Agent1 Transgress Credits
TAG_HIT	0x1F	0-3	Tag Hit
TAG_MISS	0x4B	0-3	Tag Miss
TGR_AD_CREDITS	0x2E	0-3	Number AD Ingress Credits
TGR_BL_CREDITS	0x2F	0-3	Number BL Ingress Credits
TxC_AD	0x6	0-3	AD Egress (to CMS)
TxC_BL_CREDITS	0x10	0-3	BL Egress (to CMS) Credits
TxC_EGRESS_AK	0xA	0-3	AK Egress (to CMS)
TxR_HORZ_ADS_USED	0xA6		CMS Horizontal ADS Used
TxR_HORZ_BYPASS	0xA7		CMS Horizontal Bypass Used
TxR_HORZ_CYCLES_FULL	0xA2		Cycles CMS Horizontal Egress Queue is Full
TxR_HORZ_CYCLES_NE	0xA3		Cycles CMS Horizontal Egress Queue is Not Empty
TxR_HORZ_INSERTS	0xA1		CMS Horizontal Egress Inserts
TxR_HORZ_NACK	0xA4		CMS Horizontal Egress NACKs
TxR_HORZ_OCCUPANCY	0xA0		CMS Horizontal Egress Occupancy

Table 2-235. Directly Measured M2M Box Events (Sheet 4 of 4)

Symbol Name	Event Code	Ctrs	Description
TxR_HORZ_STARVED	0xA5		CMS Horizontal Egress Injection Starvation
TxR_VERT_ADS_USED	0x9C		CMS Vertical ADS Used
TxR_VERT_BYPASS	0x9D		CMS Vertical ADS Used
TxR_VERT_BYPASS_1	0x9E		CMS Vertical ADS Used
TxR_VERT_CYCLES_FULL0	0x94		Cycles CMS Vertical Egress Queue Is Full
TxR_VERT_CYCLES_FULL1	0x95		Cycles CMS Vertical Egress Queue Is Full
TxR_VERT_CYCLES_NE0	0x96		Cycles CMS Vertical Egress Queue Is Not Empty
TxR_VERT_CYCLES_NE1	0x97		Cycles CMS Vertical Egress Queue Is Not Empty
TxR_VERT_INSERTS0	0x92		CMS Vert Egress Allocations
TxR_VERT_INSERTS1	0x93		CMS Vert Egress Allocations
TxR_VERT_IRADS_USED	0x9F		
TxR_VERT_NACK0	0x98		CMS Vertical Egress NACKs
TxR_VERT_NACK1	0x99		CMS Vertical Egress NACKs
TxR_VERT_OCCUPANCY0	0x90		CMS Vert Egress Occupancy
TxR_VERT_OCCUPANCY1	0x91		CMS Vert Egress Occupancy
TxR_VERT_STARVED0	0x9A		CMS Vertical Egress Injection Starvation
TxR_VERT_STARVED1	0x9B		CMS Vertical Egress Injection Starvation
VERT_RING_AD_IN_USE	0xB0		Vertical AD Ring In Use
VERT_RING_AKC_IN_USE	0xB4		Vertical AKC Ring In Use
VERT_RING_AK_IN_USE	0xB1		Vertical AK Ring In Use
VERT_RING_BL_IN_USE	0xB2		Vertical BL Ring in Use
VERT_RING_IV_IN_USE	0xB3		Vertical IV Ring in Use
VERT_RING_TGC_IN_USE	0xB5		Vertical TGC Ring In Use

2.7.5 M2M Box Performance Monitor Event List

The section enumerates 5th Gen Intel® Xeon® Scalable Processor performance monitoring events for the M2M box.

AG0_AD_CRD_ACQUIRED0

- **Title:**
- **Category:** Transgress Credit Events
- **Event Code:** 0x80
- **Register Restrictions :**
- **Definition:** Number of CMS agent 0 AD credits acquired in a given cycle, per transgress.

Table 2-236. Unit Masks for AG0_AD_CRD_ACQUIRED0

Extension	umask [15:8]	xtra [57:32]	Description
TGR0	bxxxxxxx1	bx1xxxxxxx xxxxxxxxxx xxxxxx	For Transgress 0
TGR1	bxxxxxx1x	bx1xxxxxxx xxxxxxxxxx xxxxxx	For Transgress 1
TGR2	bxxxxx1xx	bx1xxxxxxx xxxxxxxxxx xxxxxx	For Transgress 2
TGR3	bxxxx1xxx	bx1xxxxxxx xxxxxxxxxx xxxxxx	For Transgress 3
TGR4	bxxx1xxxx	bx1xxxxxxx xxxxxxxxxx xxxxxx	For Transgress 4
TGR5	bxx1xxxxx	bx1xxxxxxx xxxxxxxxxx xxxxxx	For Transgress 5
TGR6	bx1xxxxxx	bx1xxxxxxx xxxxxxxxxx xxxxxx	For Transgress 6
TGR7	b1xxxxxxx	bx1xxxxxxx xxxxxxxxxx xxxxxx	For Transgress 7

AG0_AD_CRD_ACQUIRED1

- **Title:**
- **Category:** Transgress Credit Events
- **Event Code:** 0x81
- **Register Restrictions :**
- **Definition:** Number of CMS agent 0 AD credits acquired in a given cycle, per transgress.

Table 2-237. Unit Masks for AG0_AD_CRD_ACQUIRED1 (Sheet 1 of 2)

Extension	umask [15:8]	xtra [57:32]	Description
TGR8	bxxxxxxx1	bx1xxxxxxx xxxxxxxxxx xxxxxx	For Transgress 8
TGR9	bxxxxxx1x	bx1xxxxxxx xxxxxxxxxx xxxxxx	For Transgress 9
TGR10	bxxxxx1xx	bx1xxxxxxx xxxxxxxxxx xxxxxx	For Transgress 10
TGR11	bxxxx1xxx	bx1xxxxxxx xxxxxxxxxx xxxxxx	For Transgress 11
TGR12	bxxx1xxxx	bx1xxxxxxx xxxxxxxxxx xxxxxx	For Transgress 12

Table 2-237. Unit Masks for AG0_AD_CRD_ACQUIRED1 (Sheet 2 of 2)

Extension	umask [15:8]	xtra [57:32]	Description
TGR13	bxx1xxxxx	bx1xxxxxxx xxxxxxxxx xxxxxx	For Transgress 13
TGR14	bx1xxxxxx	bx1xxxxxxx xxxxxxxxx xxxxxx	For Transgress 14
TGR15	b1xxxxxxx	bx1xxxxxxx xxxxxxxxx xxxxxx	For Transgress 15

AG0_AD_CRD_OCCUPANCY0

- **Title:**
- **Category:** Transgress Credit Events
- **Event Code:** 0x82
- **Register Restrictions :**
- **Definition:** Number of CMS agent 0 AD credits in use in a given cycle, per transgress.

Table 2-238. Unit Masks for AG0_AD_CRD_OCCUPANCY0

Extension	umask [15:8]	xtra [57:32]	Description
TGR0	b00000001	bx1xxxxxxx xxxxxxxxx xxxxxx	For Transgress 0
TGR1	b00000010	bx1xxxxxxx xxxxxxxxx xxxxxx	For Transgress 1
TGR2	b00000100	bx1xxxxxxx xxxxxxxxx xxxxxx	For Transgress 2
TGR3	b00001000	bx1xxxxxxx xxxxxxxxx xxxxxx	For Transgress 3
TGR4	b00010000	bx1xxxxxxx xxxxxxxxx xxxxxx	For Transgress 4
TGR5	b00100000	bx1xxxxxxx xxxxxxxxx xxxxxx	For Transgress 5
TGR6	b01000000	bx1xxxxxxx xxxxxxxxx xxxxxx	For Transgress 6
TGR7	b10000000	bx1xxxxxxx xxxxxxxxx xxxxxx	For Transgress 7

AG0_AD_CRD_OCCUPANCY1

- **Title:**
- **Category:** Transgress Credit Events
- **Event Code:** 0x83
- **Register Restrictions :**

- **Definition:** Number of CMS agent 0 AD credits in use in a given cycle, per transgress.

Table 2-239. Unit Masks for AG0_AD_CRD_OCCUPANCY1

Extension	umask [15:8]	xtra [57:32]	Description
TGR8	b00000001	bx1xxxxxxxxxxxxxxxxxxxxx	For Transgress 8
TGR9	b00000010	bx1xxxxxxxxxxxxxxxxxxxxx	For Transgress 9
TGR10	b00000100	bx1xxxxxxxxxxxxxxxxxxxxx	For Transgress 10
TGR11	b00001000	bx1xxxxxxxxxxxxxxxxxxxxx	For Transgress 11
TGR12	b00010000	bx1xxxxxxxxxxxxxxxxxxxxx	For Transgress 12
TGR13	b00100000	bx1xxxxxxxxxxxxxxxxxxxxx	For Transgress 13
TGR14	b01000000	bx1xxxxxxxxxxxxxxxxxxxxx	For Transgress 14
TGR15	b10000000	bx1xxxxxxxxxxxxxxxxxxxxx	For Transgress 15

AG0_BL_CRD_ACQUIRED0

- **Title:**
- **Category:** Transgress Credit Events
- **Event Code:** 0x88
- **Register Restrictions :**
- **Definition:** Number of CMS agent 0 BL credits acquired in a given cycle, per transgress.

Table 2-240. Unit Masks for AG0_BL_CRD_ACQUIRED0 (Sheet 1 of 2)

Extension	umask [15:8]	xtra [57:32]	Description
TGR0	bxxxxxxx1	bx1xxxxxxxxxxxxxxxxxxxxx	For Transgress 0
TGR1	bxxxxxx1x	bx1xxxxxxxxxxxxxxxxxxxxx	For Transgress 1
TGR2	bxxxxx1xx	bx1xxxxxxxxxxxxxxxxxxxxx	For Transgress 2
TGR3	bxxxx1xxx	bx1xxxxxxxxxxxxxxxxxxxxx	For Transgress 3

Table 2-240. Unit Masks for AG0_BL_CRD_ACQUIRED0 (Sheet 2 of 2)

Extension	umask [15:8]	xtra [57:32]	Description
TGR4	bxxx1xxxx	bx1xxxxxxxxxxxxxxxxxxxxx	For Transgress 4
TGR5	bxx1xxxxx	bx1xxxxxxxxxxxxxxxxxxxxx	For Transgress 5
TGR6	bx1xxxxxx	bx1xxxxxxxxxxxxxxxxxxxxx	For Transgress 6
TGR7	b1xxxxxxx	bx1xxxxxxxxxxxxxxxxxxxxx	For Transgress 7

AG0_BL_CRD_ACQUIRED1

- **Title:**
- **Category:** Transgress Credit Events
- **Event Code:** 0x89
- **Register Restrictions :**
- **Definition:** Number of CMS agent 0 BL credits acquired in a given cycle, per transgress.

Table 2-241. Unit Masks for AG0_BL_CRD_ACQUIRED1

Extension	umask [15:8]	xtra [57:32]	Description
TGR8	bxxxxxxx1	bx1xxxxxxxxxxxxxxxxxxxxx	For Transgress 8
TGR9	bxxxxxx1x	bx1xxxxxxxxxxxxxxxxxxxxx	For Transgress 9
TGR10	bxxxxx1xx	bx1xxxxxxxxxxxxxxxxxxxxx	For Transgress 10
TGR11	bxxxx1xxx	bx1xxxxxxxxxxxxxxxxxxxxx	For Transgress 11
TGR12	bxxx1xxxx	bx1xxxxxxxxxxxxxxxxxxxxx	For Transgress 12
TGR13	bxx1xxxxx	bx1xxxxxxxxxxxxxxxxxxxxx	For Transgress 13
TGR14	bx1xxxxxx	bx1xxxxxxxxxxxxxxxxxxxxx	For Transgress 14
TGR15	b1xxxxxxx	bx1xxxxxxxxxxxxxxxxxxxxx	For Transgress 15

AG0_BL_CRD_OCCUPANCY0

- **Title:**
- **Category:** Transgress Credit Events
- **Event Code:** 0x8A
- **Register Restrictions :**
- **Definition:** Number of CMS agent 0 BL credits in use in a given cycle, per transgress.

Table 2-242. Unit Masks for AG0_BL_CRD_OCCUPANCY0

Extension	umask [15:8]	xtra [57:32]	Description
TGR0	b00000001	bx1xxxxxxxxxxxxxxxxxxxxxxx	For Transgress 0
TGR1	b00000010	bx1xxxxxxxxxxxxxxxxxxxxxxx	For Transgress 1
TGR2	b00000100	bx1xxxxxxxxxxxxxxxxxxxxxxx	For Transgress 2
TGR3	b00001000	bx1xxxxxxxxxxxxxxxxxxxxxxx	For Transgress 3
TGR4	b00010000	bx1xxxxxxxxxxxxxxxxxxxxxxx	For Transgress 4
TGR5	b00100000	bx1xxxxxxxxxxxxxxxxxxxxxxx	For Transgress 5
TGR6	b01000000	bx1xxxxxxxxxxxxxxxxxxxxxxx	For Transgress 6
TGR7	b10000000	bx1xxxxxxxxxxxxxxxxxxxxxxx	For Transgress 7

AG0_BL_CRD_OCCUPANCY1

- **Title:**
- **Category:** Transgress Credit Events
- **Event Code:** 0x8B
- **Register Restrictions :**
- **Definition:** Number of CMS agent 0 BL credits in use in a given cycle, per transgress.

Table 2-243. Unit Masks for AG0_BL_CRD_OCCUPANCY1 (Sheet 1 of 2)

Extension	umask [15:8]	xtra [57:32]	Description
TGR8	b00000001	bx1xxxxxxxxxxxxxxxxxxxxxxx	For Transgress 8
TGR9	b00000010	bx1xxxxxxxxxxxxxxxxxxxxxxx	For Transgress 9

Table 2-243. Unit Masks for AG0_BL_CRD_OCCUPANCY1 (Sheet 2 of 2)

Extension	umask [15:8]	xtra [57:32]	Description
TGR10	b00000100	bx1xxxxxxx xxxxxxxxxx xxxxxx	For Transgress 10
TGR11	b00001000	bx1xxxxxxx xxxxxxxxxx xxxxxx	For Transgress 11
TGR12	b00010000	bx1xxxxxxx xxxxxxxxxx xxxxxx	For Transgress 12
TGR13	b00100000	bx1xxxxxxx xxxxxxxxxx xxxxxx	For Transgress 13
TGR14	b01000000	bx1xxxxxxx xxxxxxxxxx xxxxxx	For Transgress 14
TGR15	b10000000	bx1xxxxxxx xxxxxxxxxx xxxxxx	For Transgress 15

AG1_AD_CRD_ACQUIRED0

- **Title:**
- **Category:** Transgress Credit Events
- **Event Code:** 0x84
- **Register Restrictions :**
- **Definition:** Number of CMS agent 1 AD credits acquired in a given cycle, per transgress.

Table 2-244. Unit Masks for AG1_AD_CRD_ACQUIRED0 (Sheet 1 of 2)

Extension	umask [15:8]	xtra [57:32]	Description
TGR0	bxxxxxxx1	bx1xxxxxxx xxxxxxxxxx xxxxxx	For Transgress 0
TGR1	bxxxxxx1x	bx1xxxxxxx xxxxxxxxxx xxxxxx	For Transgress 1
TGR2	bxxxxx1xx	bx1xxxxxxx xxxxxxxxxx xxxxxx	For Transgress 2
TGR3	bxxxx1xxx	bx1xxxxxxx xxxxxxxxxx xxxxxx	For Transgress 3
TGR4	bxxx1xxxx	bx1xxxxxxx xxxxxxxxxx xxxxxx	For Transgress 4
TGR5	bxx1xxxxx	bx1xxxxxxx xxxxxxxxxx xxxxxx	For Transgress 5

Table 2-244. Unit Masks for AG1_AD_CRD_ACQUIRED0 (Sheet 2 of 2)

Extension	umask [15:8]	xtra [57:32]	Description
TGR6	bx1xxxxxx	bx1xxxxxxx xxxxxxxxxx xxxxxx	For Transgress 6
TGR7	b1xxxxxxx	bx1xxxxxxx xxxxxxxxxx xxxxxx	For Transgress 7

AG1_AD_CRD_ACQUIRED1

- **Title:**
- **Category:** Transgress Credit Events
- **Event Code:** 0x85
- **Register Restrictions :**
- **Definition:** Number of CMS agent 1 AD credits acquired in a given cycle, per transgress.

Table 2-245. Unit Masks for AG1_AD_CRD_ACQUIRED1

Extension	umask [15:8]	xtra [57:32]	Description
TGR8	bxxxxxxx1	bx1xxxxxxx xxxxxxxxxx xxxxxx	For Transgress 8
TGR9	bxxxxxx1x	bx1xxxxxxx xxxxxxxxxx xxxxxx	For Transgress 9
TGR10	bxxxxx1xx	bx1xxxxxxx xxxxxxxxxx xxxxxx	For Transgress 10
TGR11	bxxxx1xxx	bx1xxxxxxx xxxxxxxxxx xxxxxx	For Transgress 11
TGR12	bxxx1xxxx	bx1xxxxxxx xxxxxxxxxx xxxxxx	For Transgress 12
TGR13	bxx1xxxxx	bx1xxxxxxx xxxxxxxxxx xxxxxx	For Transgress 13
TGR14	bx1xxxxxx	bx1xxxxxxx xxxxxxxxxx xxxxxx	For Transgress 14
TGR15	b1xxxxxxx	bx1xxxxxxx xxxxxxxxxx xxxxxx	For Transgress 15

AG1_AD_CRD_OCCUPANCY0

- **Title:**
- **Category:** Transgress Credit Events
- **Event Code:** 0x86
- **Register Restrictions :**
- **Definition:** Number of CMS agent 1 AD credits in use in a given cycle, per transgress

Table 2-246. Unit Masks for AG1_AD_CRD_OCCUPANCY0

Extension	umask [15:8]	xtra [57:32]	Description
TGR0	b00000001	bx1xxxxxxx xxxxxxxxxx xxxxxx	For Transgress 0
TGR1	b00000010	bx1xxxxxxx xxxxxxxxxx xxxxxx	For Transgress 1
TGR2	b00000100	bx1xxxxxxx xxxxxxxxxx xxxxxx	For Transgress 2
TGR3	b00001000	bx1xxxxxxx xxxxxxxxxx xxxxxx	For Transgress 3
TGR4	b00010000	bx1xxxxxxx xxxxxxxxxx xxxxxx	For Transgress 4
TGR5	b00100000	bx1xxxxxxx xxxxxxxxxx xxxxxx	For Transgress 5
TGR6	b01000000	bx1xxxxxxx xxxxxxxxxx xxxxxx	For Transgress 6
TGR7	b10000000	bx1xxxxxxx xxxxxxxxxx xxxxxx	For Transgress 7

AG1_AD_CRD_OCCUPANCY1

- **Title:**
- **Category:** Transgress Credit Events
- **Event Code:** 0x87
- **Register Restrictions :**
- **Definition:** Number of CMS agent 1 AD credits in use in a given cycle, per transgress.

Table 2-247. Unit Masks for AG1_AD_CRD_OCCUPANCY1 (Sheet 1 of 2)

Extension	umask [15:8]	xtra [57:32]	Description
TGR8	b00000001	bx1xxxxxxx xxxxxxxxxx xxxxxx	For Transgress 8
TGR9	b00000010	bx1xxxxxxx xxxxxxxxxx xxxxxx	For Transgress 9
TGR10	b00000100	bx1xxxxxxx xxxxxxxxxx xxxxxx	For Transgress 10
TGR11	b00001000	bx1xxxxxxx xxxxxxxxxx xxxxxx	For Transgress 11
TGR12	b00010000	bx1xxxxxxx xxxxxxxxxx xxxxxx	For Transgress 12

Table 2-247. Unit Masks for AG1_AD_CRD_OCCUPANCY1 (Sheet 2 of 2)

Extension	umask [15:8]	xtra [57:32]	Description
TGR13	b00100000	bx1xxxxxxxxxxxxxxxxxxxxxxx	For Transgress 13
TGR14	b01000000	bx1xxxxxxxxxxxxxxxxxxxxxxx	For Transgress 14
TGR15	b10000000	bx1xxxxxxxxxxxxxxxxxxxxxxx	For Transgress 15

AG1_BL_CRD_ACQUIRED0

- **Title:**
- **Category:** Transgress Credit Events
- **Event Code:** 0x8C
- **Register Restrictions :**
- **Definition:** Number of CMS agent 1 BL credits acquired in a given cycle, per transgress.

Table 2-248. Unit Masks for AG1_BL_CRD_ACQUIRED0

Extension	umask [15:8]	xtra [57:32]	Description
TGR0	bxxxxxxx1	bx1xxxxxxxxxxxxxxxxxxxxxxx	For Transgress 0
TGR1	bxxxxxx1x	bx1xxxxxxxxxxxxxxxxxxxxxxx	For Transgress 1
TGR2	bxxxxx1xx	bx1xxxxxxxxxxxxxxxxxxxxxxx	For Transgress 2
TGR3	bxxxx1xxx	bx1xxxxxxxxxxxxxxxxxxxxxxx	For Transgress 3
TGR4	bxxx1xxxx	bx1xxxxxxxxxxxxxxxxxxxxxxx	For Transgress 4
TGR5	bxx1xxxxx	bx1xxxxxxxxxxxxxxxxxxxxxxx	For Transgress 5
TGR6	bx1xxxxxx	bx1xxxxxxxxxxxxxxxxxxxxxxx	For Transgress 4
TGR7	b1xxxxxxx	bx1xxxxxxxxxxxxxxxxxxxxxxx	For Transgress 5

AG1_BL_CRD_ACQUIRED1

- **Title:**
- **Category:** Transgress Credit Events
- **Event Code:** 0x8D
- **Register Restrictions :**

- **Definition:** Number of CMS agent 1 BL credits acquired in a given cycle, per transgress.

Table 2-249. Unit Masks for AG1_BL_CRD_ACQUIRED1

Extension	umask [15:8]	xtra [57:32]	Description
TGR8	bxxxxxxx1	bx1xxxxxxxxxxxxxxxxxxxxxxx	For Transgress 8
TGR9	bxxxxxx1x	bx1xxxxxxxxxxxxxxxxxxxxxxx	For Transgress 9
TGR10	bxxxxx1xx	bx1xxxxxxxxxxxxxxxxxxxxxxx	For Transgress 10
TGR11	bxxxx1xxx	bx1xxxxxxxxxxxxxxxxxxxxxxx	For Transgress 11
TGR12	bxxx1xxxx	bx1xxxxxxxxxxxxxxxxxxxxxxx	For Transgress 12
TGR13	bxx1xxxxx	bx1xxxxxxxxxxxxxxxxxxxxxxx	For Transgress 13
TGR14	bx1xxxxxx	bx1xxxxxxxxxxxxxxxxxxxxxxx	For Transgress 14
TGR15	b1xxxxxxx	bx1xxxxxxxxxxxxxxxxxxxxxxx	For Transgress 15

AG1_BL_CRD_OCCUPANCY0

- **Title:**
- **Category:** Transgress Credit Events
- **Event Code:** 0x8E
- **Register Restrictions :**
- **Definition:** Number of CMS agent 1 BL credits in use in a given cycle, per transgress.

Table 2-250. Unit Masks for AG1_BL_CRD_OCCUPANCY0 (Sheet 1 of 2)

Extension	umask [15:8]	xtra [57:32]	Description
TGR0	b00000001	bx1xxxxxxxxxxxxxxxxxxxxxxx	For Transgress 0
TGR1	b00000010	bx1xxxxxxxxxxxxxxxxxxxxxxx	For Transgress 1
TGR2	b00000100	bx1xxxxxxxxxxxxxxxxxxxxxxx	For Transgress 2
TGR3	b00001000	bx1xxxxxxxxxxxxxxxxxxxxxxx	For Transgress 3

Table 2-250. Unit Masks for AG1_BL_CRD_OCCUPANCY0 (Sheet 2 of 2)

Extension	umask [15:8]	xtra [57:32]	Description
TGR4	b00010000	bx1xxxxxxx xxxxxxxxxx xxxxxx	For Transgress 4
TGR5	b00100000	bx1xxxxxxx xxxxxxxxxx xxxxxx	For Transgress 5
TGR6	b01000000	bx1xxxxxxx xxxxxxxxxx xxxxxx	For Transgress 6
TGR7	b10000000	bx1xxxxxxx xxxxxxxxxx xxxxxx	For Transgress 7

AG1_BL_CRD_OCCUPANCY1

- **Title:**
- **Category:** Transgress Credit Events
- **Event Code:** 0x8F
- **Register Restrictions :**
- **Definition:** Number of CMS agent 1 BL credits in use in a given cycle, per transgress.

Table 2-251. Unit Masks for AG1_BL_CRD_OCCUPANCY1

Extension	umask [15:8]	xtra [57:32]	Description
TGR8	b00000001	bx1xxxxxxx xxxxxxxxxx xxxxxx	For Transgress 8
TGR9	b00000010	bx1xxxxxxx xxxxxxxxxx xxxxxx	For Transgress 9
TGR10	b00000100	bx1xxxxxxx xxxxxxxxxx xxxxxx	For Transgress 10
TGR11	b00001000	bx1xxxxxxx xxxxxxxxxx xxxxxx	For Transgress 11
TGR12	b00010000	bx1xxxxxxx xxxxxxxxxx xxxxxx	For Transgress 12
TGR13	b00100000	bx1xxxxxxx xxxxxxxxxx xxxxxx	For Transgress 13
TGR14	b01000000	bx1xxxxxxx xxxxxxxxxx xxxxxx	For Transgress 14
TGR15	b10000000	bx1xxxxxxx xxxxxxxxxx xxxxxx	For Transgress 15

BYPASS_M2M_EGRESS

- **Title:**
- **Category:** BL Egress Events
- **Event Code:** 0x15
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-252. Unit Masks for BYPASS_M2M_EGRESS

Extension	umask [15:8]	Description
TAKEN	bxxxxxxx1	Taken
NOT_TAKEN	bxxxxxx1x	Not Taken

CLOCKTICKS

- **Title:**
- **Category:** Clock ticks Events
- **Event Code:** 0x1
- **Register Restrictions :** 0-3
- **Definition:**

CMS_CLOCKTICKS

- **Title:**
- **Category:** Misc Events
- **Event Code:** 0xC0
- **Register Restrictions :**
- **Definition:**

DIRECT2CORE_NOT_TAKEN_DIRSTATE

- **Title:**
- **Category:** Direct To Core Events
- **Event Code:** 0x17
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-253. Unit Masks for DIRECT2CORE_NOT_TAKEN_DIRSTATE

Extension	umask [15:8]	Description
EGRESS	bxxxxxxx1	Egress Counts the number of time D2C was not honored by egress due to directory state constraints
NON_CISGRESS	bxxxxxx1x	Non Cisgress Counts the number of time non cisgress D2C was not honored by egress due to directory state constraints

DIRECT2CORE_NOT_TAKEN_NOTFORKED

- **Title:**
- **Category:** Direct To Core Events
- **Event Code:** 0x4A
- **Register Restrictions :** 0-3
- **Definition:**

DIRECT2CORE_TXN_OVERRIDE

- **Title:**
- **Category:** Direct To Core Events
- **Event Code:** 0x18
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-254. Unit Masks for DIRECT2CORE_TXN_OVERRIDE

Extension	umask [15:8]	Description
PMM_HIT	bxxxxxxx1	2LM Hit
CISGRESS	bxxxxxx1x	Cisgress

DIRECT2UPI_NOT_TAKEN_CREDITS

- **Title:**
- **Category:** Direct to UPI Events
- **Event Code:** 0x1B
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-255. Unit Masks for DIRECT2UPI_NOT_TAKEN_CREDITS

Extension	umask [15:8]	Description
EGRESS	bxxxxxxx1	All Counts the number of d2k was not done due to credit constraints
NON_CISGRESS	bxxxxxx1x	Non Cisgress Counts the number of non Cisgress d2k wasn't done due to credit constraints
CISGRESS	bxxxxx1xx	Cisgress Counts the number of Cisgress d2k wasn't done due to credit constraints

DIRECT2UPI_NOT_TAKEN_DIRSTATE

- **Title:**
- **Category:** Direct to UPI Events
- **Event Code:** 0x1A
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-256. Unit Masks for DIRECT2UPI_NOT_TAKEN_DIRSTATE

Extension	umask [15:8]	Description
EGRESS	bxxxxxxx1	Egress Ignored D2U Counts the number of time D2K was not honored by egress due to directory state constraints
NON_CISGRESS	bxxxxxx1x	Non Cisgress D2U Ignored Counts non Cisgress d2K that was not honored due to directory constraints
CISGRESS	bxxxxx1xx	Cisgress D2U Ignored Counts Cisgress d2K that was not honored due to directory constraints

DIRECT2UPI_TXN_OVERRIDE

- **Title:**
- **Category:** Direct to UPI Events
- **Event Code:** 0x1C
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-257. Unit Masks for DIRECT2UPI_TXN_OVERRIDE

Extension	umask [15:8]	Description
PMM_HIT	bxxxxxxx1	Counts the number of times D2K was not honored even though the incoming request had d2k set
CISGRESS	bxxxxxx1x	

DIRECTORY_HIT

- **Title:**
- **Category:** Directory State Events
- **Event Code:** 0x1D
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-258. Unit Masks for DIRECTORY_HIT (Sheet 1 of 2)

Extension	umask [15:8]	Description
DIRTY_I	bxxxxxxx1	On Dirty Line in I State Counts hit Txn with directory=I and metadata[3] is dirty
DIRTY_S	bxxxxxx1x	On Dirty Line in S State Counts hit Txn with directory=S and metadata[3] is dirty
DIRTY_P	bxxxxx1xx	On Dirty Line in L State Counts hit Txn with directory=P and metadata[3] is dirty
DIRTY_A	bxxxx1xxx	On Dirty Line in A State Counts hit Txn with directory=A and metadata[3] is dirty
CLEAN_I	bxxx1xxxx	On NonDirty Line in I State Counts hit Txn with directory=I and metadata[3] is clean

Table 2-258. Unit Masks for DIRECTORY_HIT (Sheet 2 of 2)

Extension	umask [15:8]	Description
CLEAN_S	bxx1xxxxx	On NonDirty Line in S State Counts hit Txn with directory=S and metadata[3] is clean
CLEAN_P	bx1xxxxxx	On NonDirty Line in L State Counts hit Txn with directory=P and metadata[3] is clean
CLEAN_A	b1xxxxxxx	On NonDirty Line in A State Counts hit Txn with directory=A and metadata[3] is clean

DIRECTORY_LOOKUP

- **Title:**
- **Category:** Directory Events
- **Event Code:** 0x20
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-259. Unit Masks for DIRECTORY_LOOKUP

Extension	umask [15:8]	Description
ANY	bxxxxxxx1	Found in any state Counts the number of hit data returns to egress with any directory to non persistent memory
STATE_I	bxxxxxx1x	Found in I state Counts the number of hit data returns to egress with directory I to non persistent memory
STATE_S	bxxxxx1xx	Found in S state Counts the number of hit data returns to egress with directory S to non persistent memory
STATE_A	bxxxx1xxx	Found in A state Counts the number of hit data returns to egress with directory A to non persistent memory

DIRECTORY_MISS

- **Title:**
- **Category:** Directory State Events
- **Event Code:** 0x1E
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-260. Unit Masks for DIRECTORY_MISS (Sheet 1 of 2)

Extension	umask [15:8]	Description
DIRTY_I	bxxxxxxx1	On Dirty Line in I State Counts miss Txn with directory=I and metadata[3] is dirty
DIRTY_S	bxxxxxx1x	On Dirty Line in S State Counts miss Txn with directory=S and metadata[3] is dirty
DIRTY_P	bxxxxx1xx	On Dirty Line in L State Counts miss Txn with directory=P and metadata[3] is dirty

Table 2-260. Unit Masks for DIRECTORY_MISS (Sheet 2 of 2)

Extension	umask [15:8]	Description
DIRTY_A	bxxxx1xxx	On Dirty Line in A State Counts miss Txn with directory=A and metadata[3] is dirty
CLEAN_I	bxxx1xxxx	On NonDirty Line in I State Counts miss Txn with directory=I and metadata[3] is clean
CLEAN_S	bxx1xxxxx	On NonDirty Line in S State Counts miss Txn with directory=S and metadata[3] is clean
CLEAN_P	bx1xxxxxx	On NonDirty Line in L State Counts miss Txn with directory=P and metadata[3] is clean
CLEAN_A	b1xxxxxxx	On NonDirty Line in A State Counts miss Txn with directory=A and metadata[3] is clean

DIRECTORY_UPDATE

- **Title:**
- **Category:** Directory Events
- **Event Code:** 0x21
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-261. Unit Masks for DIRECTORY_UPDATE (Sheet 1 of 2)

Extension	umask [15:8]	xtra [57:32]	Description
HIT_NON_PMM	bxxxxxxx1	bxxxxxxx1	Counts any 1lm or 2lm hit data return that would result in directory update to non persistent memory (DRAM or HBM)
MISS_NON_PMM	bxxxxxxx1	bxxxxxx1x	Counts any 2lm miss data return that would result in directory update to non persistent memory (DRAM or HBM)
I_TO_S_HIT_NON_PMM	bxxxxxx1x	bxxxxxxx1	Counts 1lm or 2lm hit data returns that would result in directory update from I to S to non persistent memory (DRAM or HBM)
I_TO_S_MISS_NON_PMM	bxxxxxx1x	bxxxxxx1x	Counts 2lm miss data returns that would result in directory update from I to S to non persistent memory (DRAM or HBM)
I_TO_A_HIT_NON_PMM	bxxxxx1xx	bxxxxxxx1	Counts 1lm or 2lm hit data returns that would result in directory update from I to A to non persistent memory (DRAM or HBM)
I_TO_A_MISS_NON_PMM	bxxxxx1xx	bxxxxxx1x	Counts 2lm miss data returns that would result in directory update from I to A to non persistent memory (DRAM or HBM)
S_TO_I_HIT_NON_PMM	bxxxx1xxx	bxxxxxxx1	Counts 1lm or 2lm hit data returns that would result in directory update from S to I to non persistent memory (DRAM or HBM)
S_TO_I_MISS_NON_PMM	bxxxx1xxx	bxxxxxx1x	Counts 2lm miss data returns that would result in directory update from S to I to non persistent memory (DRAM or HBM)
S_TO_A_HIT_NON_PMM	bxxx1xxxx	bxxxxxxx1	Counts 1lm or 2lm hit data returns that would result in directory update from S to A to non persistent memory (DRAM or HBM)
S_TO_A_MISS_NON_PMM	bxxx1xxxx	bxxxxxx1x	Counts 2lm miss data returns that would result in directory update from S to A to non persistent memory (DRAM or HBM)

Table 2-261. Unit Masks for DIRECTORY_UPDATE (Sheet 2 of 2)

Extension	umask [15:8]	xtra [57:32]	Description
A_TO_I_HIT_NON_PMM	bxx1xxxxx	bxxxxxxx1	Counts 1lm or 2lm hit data returns that would result in directory update from A to I to non persistent memory (DRAM or HBM)
A_TO_I_MISS_NON_PMM	bxx1xxxxx	bxxxxxx1x	Counts 2lm miss data returns that would result in directory update from A to I to non persistent memory (DRAM or HBM)
A_TO_S_HIT_NON_PMM	bx1xxxxxx	bxxxxxxx1	Counts 1lm or 2lm hit data returns that would result in directory update from A to S to non persistent memory (DRAM or HBM)
A_TO_S_MISS_NON_PMM	bx1xxxxxx	bxxxxxx1x	Counts 2lm miss data returns that would result in directory update from A to S to non persistent memory (DRAM or HBM)

EGRESS_ORDERING

- **Title:**
- **Category:** Horizontal In Use Ring Events
- **Event Code:** 0xBA
- **Register Restrictions :**
- **Definition:** Counts number of cycles IV was blocked in the TGR egress due to SNP/GO ordering requirements

Table 2-262. Unit Masks for EGRESS_ORDERING

Extension	umask [15:8]	xtra [57:32]	Description
IV_SNOOPGO_UP	bxxxxxxx1	bx1xxxxxx xxxxxxxxx xxxxxxxx	Up
IV_SNOOPGO_DN	bxxxxx1xx	bx1xxxxxx xxxxxxxxx xxxxxxxx	Down

HORZ_RING_AD_IN_USE

- **Title:**
- **Category:** Horizontal In Use Ring Events
- **Event Code:** 0xB6
- **Register Restrictions :**
- **Definition:** Counts the number of cycles that the horizontal AD ring is being used at this ring stop. This includes when packets are passing by and when packets are being sunk, but does not include when packets are being sent from the ring stop. We really have two rings -- a clockwise ring and a counter-clockwise ring. On the left side of the ring, the "UP" direction is on the clockwise ring and "DN" is on the counter-clockwise ring. On the right side of the ring, this is reversed. The first half of the CBos are on the left side of the ring, and the second half are on the right side of the ring. In other words (for example), in a 4c part, Cbo 0 UP AD is NOT the same ring as CBo 2 UP AD because they are on opposite sides of the ring.

Table 2-263. Unit Masks for HORZ_RING_AD_IN_USE

Extension	umask [15:8]	xtra [57:32]	Description
LEFT_EVEN	bxxxxxxx1	bx1xxxxxxxxxxxxxxxxxxxxxxx	Left and Even
LEFT_ODD	bxxxxxx1x	bx1xxxxxxxxxxxxxxxxxxxxxxx	Left and Odd
RIGHT_EVEN	bxxxxx1xx	bx1xxxxxxxxxxxxxxxxxxxxxxx	Right and Even
RIGHT_ODD	bxxxx1xxx	bx1xxxxxxxxxxxxxxxxxxxxxxx	Right and Odd

HORZ_RING_AKC_IN_USE

- **Title:**
- **Category:** Horizontal In Use Ring Events
- **Event Code:** 0xBB
- **Register Restrictions :**
- **Definition:** Counts the number of cycles that the horizontal AKC ring is being used at this ring stop. This includes when packets are passing by and when packets are being sunk, but does not include when packets are being sent from the ring stop.

Table 2-264. Unit Masks for HORZ_RING_AKC_IN_USE

Extension	umask [15:8]	xtra [57:32]	Description
LEFT_EVEN	bxxxxxxx1	bx1xxxxxxxxxxxxxxxxxxxxxxx	Left and Even
LEFT_ODD	bxxxxxx1x	bx1xxxxxxxxxxxxxxxxxxxxxxx	Left and Odd
RIGHT_EVEN	bxxxxx1xx	bx1xxxxxxxxxxxxxxxxxxxxxxx	Right and Even
RIGHT_ODD	bxxxx1xxx	bx1xxxxxxxxxxxxxxxxxxxxxxx	Right and Odd

HORZ_RING_AK_IN_USE

- **Title:**
- **Category:** Horizontal In Use Ring Events
- **Event Code:** 0xB7
- **Register Restrictions :**
- **Definition:** Counts the number of cycles that the horizontal AK ring is being used at this ring stop. This includes when packets are passing by and when packets are being sunk, but does not include when packets are being sent from the ring stop. We really have two rings -- a clockwise ring and a counter-clockwise ring. On the left side of the ring, the "UP" direction is on the clockwise ring and "DN" is on the counter-clockwise ring. On the right side of the ring, this is reversed. The first half of the CBos are on the left side of the ring, and the second half are on the right side of the ring.

In other words (for example), in a 4c part, Cbo 0 UP AD is NOT the same ring as CBo 2 UP AD because they are on opposite sides of the ring.

Table 2-265. Unit Masks for HORZ_RING_AK_IN_USE

Extension	umask [15:8]	xtra [57:32]	Description
LEFT_EVEN	bxxxxxxx1		Left and Even
LEFT_ODD	bxxxxxx1x	bx1xxxxxx xxxxxxxxx xxxxxxxx	Left and Odd
RIGHT_EVEN	bxxxxx1xx	bx1xxxxxx xxxxxxxxx xxxxxxxx	Right and Even
RIGHT_ODD	bxxxx1xxx	bx1xxxxxx xxxxxxxxx xxxxxxxx	Right and Odd

HORZ_RING_BL_IN_USE

- **Title:**
- **Category:** Horizontal In Use Ring Events
- **Event Code:** 0xB8
- **Register Restrictions :**
- **Definition:** Counts the number of cycles that the horizontal BL ring is being used at this ring stop. This includes when packets are passing by and when packets are being sunk, but does not include when packets are being sent from the ring stop. We really have two rings -- a clockwise ring and a counter-clockwise ring. On the left side of the ring, the "UP" direction is on the clockwise ring and "DN" is on the counter-clockwise ring. On the right side of the ring, this is reversed. The first half of the CBos are on the left side of the ring, and the second half are on the right side of the ring. In other words (for example), in a 4c part, Cbo 0 UP AD is NOT the same ring as CBo 2 UP AD because they are on opposite sides of the ring.

Table 2-266. Unit Masks for HORZ_RING_BL_IN_USE

Extension	umask [15:8]	xtra [57:32]	Description
LEFT_EVEN	bxxxxxxx1	bx1xxxxxx xxxxxxxxx xxxxxxxx	Left and Even
LEFT_ODD	bxxxxxx1x	bx1xxxxxx xxxxxxxxx xxxxxxxx	Left and Odd
RIGHT_EVEN	bxxxxx1xx	bx1xxxxxx xxxxxxxxx xxxxxxxx	Right and Even
RIGHT_ODD	bxxxx1xxx	bx1xxxxxx xxxxxxxxx xxxxxxxx	Right and Odd

HORZ_RING_IV_IN_USE

- **Title:**
- **Category:** Horizontal In Use Ring Events
- **Event Code:** 0xB9
- **Register Restrictions :**

- **Definition:** Counts the number of cycles that the horizontal IV ring is being used at this ring stop. This includes when packets are passing by and when packets are being sunk, but does not include when packets are being sent from the ring stop. There is only one IV ring. Therefore, if one wants to monitor the "Even" ring, they should select both UP_EVEN and DN_EVEN. To monitor the "Odd" ring, they should select both UP_ODD and DN_ODD.

Table 2-267. Unit Masks for HORZ_RING_IV_IN_USE

Extension	umask [15:8]	xtra [57:32]	Description
LEFT	bxxxxxxx1	bx1xxxxxx xxxxxxxxx xxxxxxxx	Left
RIGHT	bxxxxx1xx	bx1xxxxxx xxxxxxxxx xxxxxxxx	Right

IMC_READS

- **Title:**
- **Category:** iMC Events
- **Event Code:** 0x24
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-268. Unit Masks for IMC_READS (Sheet 1 of 2)

Extension	umask [15:8]	xtra [57:32]	Description
CH0_NORMAL	bxxxxxxx1	bxxxxxxx1	
CH1_NORMAL	bxxxxxxx1	bxxxxxx1x	
NORMAL	bxxxxxxx1	b011	
CH0_ISOCH	bxxxxxx1x	bxxxxxxx1	
CH1_ISOCH	bxxxxxx1x	bxxxxxx1x	
ISOCH	bxxxxxx1x	b011	
ALL	bxxxxx1xx	b011	
CH0_ALL	bxxxxx1xx	bxxxxxxx1	
CH1_ALL	bxxxxx1xx	bxxxxxx1x	
CH0_TO_DDR_AS_MEM	bxxxx1xxx	bxxxxxxx1	
CH1_TO_DDR_AS_MEM	bxxxx1xxx	bxxxxxx1x	
TO_DDR_AS_MEM	bxxxx1xxx	b011	
CH0_TO_DDR_AS_CACHE	bxxx1xxxx	bxxxxxxx1	

Table 2-268. Unit Masks for IMC_READS (Sheet 2 of 2)

Extension	umask [15:8]	xtra [57:32]	Description
CH1_TO_DDR_AS_CACHE	bxxx1xxxx	bxxxxxx1x	
TO_DDR_AS_CACHE	bxxx1xxxx	b011	
CH0_TO_PMM	bxx1xxxxx	bxxxxxxx1	
CH1_TO_PMM	bxx1xxxxx	bxxxxxx1x	
TO_PMM	bxx1xxxxx	b011	
CH0_FROM_TGR	bx1xxxxxx	bxxxxxxx1	
CH1_FROM_TGR	bx1xxxxxx	bxxxxxx1x	
FROM_TGR	bx1xxxxxx	b011	

IMC_WRITES

- **Title:**
- **Category:** iMC Events
- **Event Code:** 0x25
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-269. Unit Masks for IMC_WRITES (Sheet 1 of 2)

Extension	umask [15:8]	xtra [57:32]	Description
CH0_FROM_TGR	bxxxxxxxx	bxxxx1xx1	From TGR - Ch0
CH0_NI	bxxxxxxxx	bxxxx1x1x	Non-Inclusive - Ch0
CH0_NI_MISS	bxxxxxxxx	bxxxx11xx	Non-Inclusive Miss - Ch0
CH1_FROM_TGR	bxxxxxxxx	bxxx1xxx1	From TGR - Ch1
CH1_NI	bxxxxxxxx	bxxx1xx1x	Non-Inclusive - Ch1
CH1_NI_MISS	bxxxxxxxx	bxxx1x1xx	Non-Inclusive Miss - Ch1
FROM_TGR	bxxxxxxxx	bxxx11xx1	From TGR - All Channels
NI	bxxxxxxxx	bxxx11x1x	Non-Inclusive - All Channels
NI_MISS	bxxxxxxxx	bxxx111xx	Non-Inclusive Miss - All Channels
CH0_FULL	bxxxxxxx1	bxxxx1xxx	
CH1_FULL	bxxxxxxx1	bxxx1xxxx	Full Line Non-ISOCH - Ch1

Table 2-269. Unit Masks for IMC_WRITES (Sheet 2 of 2)

Extension	umask [15:8]	xtra [57:32]	Description
FULL	bxxxxxxx1	bxxx11xxx	Full Non-ISOCH - All Channels
CH0_PARTIAL	bxxxxxx1x	bxxxx1xxx	
CH1_PARTIAL	bxxxxxx1x	bxxx1xxxx	Partial Non-ISOCH - Ch1
PARTIAL	bxxxxxx1x	bxxx11xxx	Partial Non-ISOCH - All Channels
CH0_FULL_ISOCH	bxxxxx1xx	bxxxx1xxx	
CH1_FULL_ISOCH	bxxxxx1xx	bxxx1xxxx	ISOCH Full Line - Ch1
FULL_ISOCH	bxxxxx1xx	bxxx11xxx	ISOCH Full Line - All Channels
CH0_PARTIAL_ISOCH	bxxxx1xxx	bxxxx1xxx	
CH1_PARTIAL_ISOCH	bxxxx1xxx	bxxx1xxxx	ISOCH Partial - Ch1
PARTIAL_ISOCH	bxxxx1xxx	bxxx11xxx	ISOCH Partial - All Channels
ALL	bxxx1xxxx	bxxx11xxx	All Writes - All Channels
CH0_ALL	bxxx1xxxx	bxxxx1xxx	
CH1_ALL	bxxx1xxxx	bxxx1xxxx	All Writes - Ch1
CH0_TO_DDR_AS_MEM	bxx1xxxxx	bxxxx1xxx	
CH1_TO_DDR_AS_MEM	bxx1xxxxx	bxxx1xxxx	DDR - Ch1
TO_DDR_AS_MEM	bxx1xxxxx	bxxx11xxx	DDR - All Channels
CH0_TO_DDR_AS_CACHE	bx1xxxxxx	bxxxx1xxx	DDR, acting as Cache - Ch0
CH1_TO_DDR_AS_CACHE	bx1xxxxxx	bxxx1xxxx	DDR, acting as Cache - Ch1
TO_DDR_AS_CACHE	bx1xxxxxx	bxxx11xxx	DDR, acting as Cache - All Channels
CH0_TO_PMM	b1xxxxxxx	bxxxx1xxx	PMM - Ch0 Counts all PMM DIMM writes requests (full line and partial) sent from M2M to iMC
CH1_TO_PMM	b1xxxxxxx	bxxx1xxxx	PMM - Ch1 Counts all PMM DIMM writes requests (full line and partial) sent from M2M to iMC
TO_PMM	b1xxxxxxx	bxxx11xxx	PMM - All Channels

MISC_EXTERNAL

- **Title:**
- **Category:** External Misc Events (for example, from MS2IDI)
- **Event Code:** 0xE6
- **Register Restrictions :**
- **Definition:**

Table 2-270. Unit Masks for MISC_EXTERNAL

Extension	umask [15:8]	xtra [57:32]	Description
MBE_INST0	bxxxxxxx1	bx1xxxxxx xxxxxxxxx xxxxxxxx	Number of cycles MBE is high for MS2IDI0
MBE_INST1	bxxxxxx1x	bx1xxxxxx xxxxxxxxx xxxxxxxx	Number of cycles MBE is high for MS2IDI1

Nothing

- **Title:**
- **Category:** Clock ticks Events
- **Event Code:** 0x0
- **Register Restrictions :**
- **Definition:**

PREFCAM_DEALLOCS

- **Title:**
- **Category:** Prefetch CAM Events
- **Event Code:** 0x57
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-271. Unit Masks for PREFCAM_DEALLOCS (Sheet 1 of 2)

Extension	umask [15:8]	xtra [57:32]	Description
CMS0_CH0	bxxxxxxx1	bxxxxxxx1	Chl 0 prefetch entry reset due to Add 0 port entry
CMS0_CH1	bxxxxxxx1	bxxxxxx1x	Chl 1 prefetch entry reset due to Add 0 port entry
CMS1_CH0	bxxxxxx1x	bxxxxxxx1	Chl 0 prefetch entry reset due to Add 1 port entry
CMS1_CH1	bxxxxxx1x	bxxxxxx1x	Chl 1 prefetch entry reset due to Add 1 port entry
MISS_CH0	bxxxxx1xx	bxxxxxxx1	Chl 0 prefetch entry reset due to miss invalidate
MISS_CH1	bxxxxx1xx	bxxxxxx1x	Chl 1 prefetch entry reset due to miss invalidate
RSP_CH0	bxxxx1xxx	bxxxxxxx1	Chl 0 prefetch entry reset due to pending entry reset
RSP_CH1	bxxxx1xxx	bxxxxxx1x	Chl 1 prefetch entry reset due to pending entry reset

Table 2-271. Unit Masks for PREFCAM_DEALLOCS (Sheet 2 of 2)

Extension	umask [15:8]	xtra [57:32]	Description
CRD_CH0	bxxx1xxxx	bxxxxxxx1	Chl 0 prefetch entry reset due to credit pend reset
CRD_CH1	bxxx1xxxx	bxxxxxx1x	Chl 1 prefetch entry reset due to credit pend reset
ALL_CH0	bxx1xxxxx	bxxxxxxx1	Chl 0 prefetch entry reset due to all pend reset
ALL_CH1	bxx1xxxxx	bxxxxxx1x	Chl 1 prefetch entry reset due to all pend reset

PREFCAM_DEMAND_DROPS

- **Title:**
- **Category:** Prefetch CAM Events
- **Event Code:** 0x58
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-272. Unit Masks for PREFCAM_DEMAND_DROPS

Extension	umask [15:8]	Description
CH0_XPT	bxxxxxxx1	XPT - Ch 0 Chl 0 Xpt prefetch drop
CH0_UPI	bxxxxxx1x	UPI - Ch 0 Chl 0 kti prefetch drop
CH1_XPT	bxxxxx1xx	XPT - Ch 1 Chl 1 Xpt prefetch drop
XPT_ALLCH	b00000101	XPT - All Channels
CH1_UPI	bxxxx1xxx	Intel UPI - Ch 1 Chl 1 kti prefetch drop
UPI_ALLCH	b00001010	Intel UPI - All Channels

PREFCAM_DROP_REASONS_CH0

- **Title:**
- **Category:** Prefetch CAM Events
- **Event Code:** 0x59
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-273. Unit Masks for PREFCAM_DROP_REASONS_CH0 (Sheet 1 of 2)

Extension	umask [15:8]	xtra [57:32]	Description
RPQ_PROXY	bxxxxxxxx	bxxxxxxx1	Chl 0 prefetch drop due to RPQ proxy
UPI_THRESH	bxxxxxxxx	bxxxxx1xx	Chl 0 prefetch drop due to KTI threshold

Table 2-273. Unit Masks for PREFCAM_DROP_REASONS_CH0 (Sheet 2 of 2)

Extension	umask [15:8]	xtra [57:32]	Description
XPT_THRESH	bxxxxxxxx	bxxxxxx1x	Chl 0 prefetch drop due to XPT threshold
PF_SECURE_DROP	bxxxxxxx1	bxxxxxxxx	Chl 0 prefetch drop due to secure region
NOT_PF_SAD_REGION	bxxxxxx1x	bxxxxxxxx	Chl 0 prefetch drop due to no match in prefetch SAD
PF_CAM_HIT	bxxxxx1xx	bxxxxxxxx	Chl 0 prefetch drop due to PF cam hit
STOP_B2B	bxxxx1xxx	bxxxxxxxx	Chl 0 prefetch drop due to back to back prefetch
ERRORBLK_RxC	bxxx1xxxx	bxxxxxxxx	Chl 0 prefetch drop due to error block AD ingress
PF_AD_CRD	bxx1xxxxx	bxxxxxxxx	Chl 0 prefetch drop due to AD ingress credit
PF_CAM_FULL	bx1xxxxxx	bxxxxxxxx	Chl 0 prefetch drop due to PF cam full
WPQ_PROXY	b1xxxxxxx	bxxxxxxxx	Chl 0 prefetch drop due to WPQ proxy

PREFCAM_DROP_REASONS_CH1

- **Title:**
- **Category:** Prefetch CAM Events
- **Event Code:** 0x5A
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-274. Unit Masks for PREFCAM_DROP_REASONS_CH1 (Sheet 1 of 2)

Extension	umask [15:8]	xtra [57:32]	Description
RPQ_PROXY	bxxxxxxxx	bxxxxxxx1	Chl 1 prefetch drop due to RPQ proxy
UPI_THRESH	bxxxxxxxx	bxxxxx1xx	Chl 1 prefetch drop due to KTI threshold
XPT_THRESH	bxxxxxxxx	bxxxxxx1x	Chl 1 prefetch drop due to XPT threshold
PF_SECURE_DROP	bxxxxxxx1	bxxxxxxxx	Chl 1 prefetch drop due to secure region
NOT_PF_SAD_REGION	bxxxxxx1x	bxxxxxxxx	Chl 1 prefetch drop due to no match in prefetch SAD
PF_CAM_HIT	bxxxxx1xx	bxxxxxxxx	Chl 1 prefetch drop due to PF cam hit
STOP_B2B	bxxxx1xxx	bxxxxxxxx	Chl 1 prefetch drop due to back to back prefetch
ERRORBLK_RxC	bxxx1xxxx	bxxxxxxxx	Chl 1 prefetch drop due to error block AD ingress
PF_AD_CRD	bxx1xxxxx	bxxxxxxxx	Chl 1 prefetch drop due to ad ingress credit

Table 2-274. Unit Masks for PREFCAM_DROP_REASONS_CH1 (Sheet 2 of 2)

Extension	umask [15:8]	xtra [57:32]	Description
PF_CAM_FULL	bx1xxxxxx	bxxxxxxxx	Chl 1 prefetch drop due to PF cam full
WPQ_PROXY	b1xxxxxxx	bxxxxxxxx	Chl 1 prefetch drop due to WPQ proxy

PREFCAM_INSERTS

- **Title:**
- **Category:** Prefetch CAM Events
- **Event Code:** 0x56
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-275. Unit Masks for PREFCAM_INSERTS

Extension	umask [15:8]	Description
CH0_XPT	bxxxxxxx1	XPT - Ch 0
CH0_UPI	bxxxxxx1x	UPI - Ch 0
CH1_XPT	bxxxxx1xx	XPT - Ch 1
XPT_ALLCH	b00000101	XPT - All Channels
CH1_UPI	bxxxx1xxx	UPI - Ch 1
UPI_ALLCH	b00001010	UPI - All Channels

PREFCAM_RxC_DEALLOCS

- **Title:**
- **Category:** Prefetch CAM Events
- **Event Code:** 0x62
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-276. Unit Masks for PREFCAM_RxC_DEALLOCS

Extension	umask [15:8]	Description
SQUASHED	bxxxxxxx1	Pf AD ingress dealloc due to prefetch squashed
1LM_POSTED	bxxxxxx1x	Pf AD ingress dealloc due to 1LM posted
PMM_MEMMODE_ACCEPT	bxxxxx1xx	Pf AD ingress dealloc due to 2LM accept
CIS	bxxxx1xxx	Pf AD ingress dealloc due to cisgress push

RING_BOUNCES_HORZ

- **Title:**
- **Category:** Horizontal Ring Events
- **Event Code:** 0xAC
- **Register Restrictions :**
- **Definition:** Number of cycles incoming messages from the horizontal ring that were bounced, by ring type.

Table 2-277. Unit Masks for RING_BOUNCES_HORZ

Extension	umask [15:8]	xtra [57:32]	Description
AD	bxxxxxxx1	bx1xxxxxx xxxxxxxxx xxxxxxxx	AD
AK	bxxxxxx1x	bx1xxxxxx xxxxxxxxx xxxxxxxx	AK
BL	bxxxxx1xx	bx1xxxxxx xxxxxxxxx xxxxxxxx	BL
IV	bxxxx1xxx	bx1xxxxxx xxxxxxxxx xxxxxxxx	IV

RING_BOUNCES_VERT

- **Title:**
- **Category:** Vertical Ring Events
- **Event Code:** 0xAA
- **Register Restrictions :**
- **Definition:** Number of cycles incoming messages from the vertical ring that were bounced, by ring type.

Table 2-278. Unit Masks for RING_BOUNCES_VERT

Extension	umask [15:8]	xtra [57:32]	Description
AD	bxxxxxxx1	bx1xxxxxx xxxxxxxxx xxxxxxxx	AD
AK	bxxxxxx1x	bx1xxxxxx xxxxxxxxx xxxxxxxx	Acknowledgments to core
BL	bxxxxx1xx	bx1xxxxxx xxxxxxxxx xxxxxxxx	Data Responses to core
IV	bxxxx1xxx	bx1xxxxxx xxxxxxxxx xxxxxxxx	Snoops of processors cache.
AKC	bxxx1xxxx	bx1xxxxxx xxxxxxxxx xxxxxxxx	

RING_SINK_STARVED_HORZ

- **Title:**
- **Category:** Horizontal Ring Events
- **Event Code:** 0xAD
- **Register Restrictions :**
- **Definition:**

Table 2-279. Unit Masks for RING_SINK_STARVED_HORZ

Extension	umask [15:8]	xtra [57:32]	Description
AD	bxxxxxxx1	bx1xxxxxx xxxxxxxxx xxxxxxxx	AD
AK	bxxxxxx1x	bx1xxxxxx xxxxxxxxx xxxxxxxx	AK
BL	bxxxxx1xx	bx1xxxxxx xxxxxxxxx xxxxxxxx	BL
IV	bxxxx1xxx	bx1xxxxxx xxxxxxxxx xxxxxxxx	IV
AK_AG1	bxx1xxxxx	bx1xxxxxx xxxxxxxxx xxxxxxxx	Acknowledgments to Agent 1

RING_SINK_STARVED_VERT

- **Title:**
- **Category:** Vertical Ring Events
- **Event Code:** 0xAB
- **Register Restrictions :**
- **Definition:**

Table 2-280. Unit Masks for RING_SINK_STARVED_VERT

Extension	umask [15:8]	xtra [57:32]	Description
AD	bxxxxxxx1	bx1xxxxxx xxxxxxxxx xxxxxxxx	AD
AK	bxxxxxx1x	bx1xxxxxx xxxxxxxxx xxxxxxxx	Acknowledgments to core
BL	bxxxxx1xx	bx1xxxxxx xxxxxxxxx xxxxxxxx	Data Responses to core
IV	bxxxx1xxx	bx1xxxxxx xxxxxxxxx xxxxxxxx	Snoops of processor's cache
AKC	bxxx1xxxx	bx1xxxxxx xxxxxxxxx xxxxxxxx	

RING_SRC_THRTL

- **Title:**
- **Category:** Horizontal Ring Events
- **Event Code:** 0xAE
- **Register Restrictions :**
- **Definition:**

RxR_BUSY_STARVED

- **Title:**
- **Category:** Ingress Transgress Events
- **Event Code:** 0xE5
- **Register Restrictions :**
- **Definition:** Counts cycles under injection starvation mode. This starvation is triggered when the CMS Ingress cannot send a transaction onto the mesh for a long period of time. In this case, because a message from the other queue has higher priority.

Table 2-281. Unit Masks for RxR_BUSY_STARVED

Extension	umask [15:8]	xtra [57:32]	Description
AD_UNCRD	b00000001	bx1xxxxxx xxxxxxxxx xxxxxxxx	AD - Uncredited
BL_UNCRD	b00000100	bx1xxxxxx xxxxxxxxx xxxxxxxx	BL - Uncredited
AD_CRD	b00010000	bx1xxxxxx xxxxxxxxx xxxxxxxx	AD - Credited
AD_ALL	b00010001	bx1xxxxxx xxxxxxxxx xxxxxxxx	AD - All All == Credited + Uncredited
BL_CRD	b01000000	bx1xxxxxx xxxxxxxxx xxxxxxxx	BL - Credited
BL_ALL	b01000100	bx1xxxxxx xxxxxxxxx xxxxxxxx	BL - All All == Credited + Uncredited

RxR_BYPASS

- **Title:**
- **Category:** Ingress Transgress Events
- **Event Code:** 0xE2
- **Register Restrictions :**
- **Definition:** Number of packets bypassing the CMS ingress.

Table 2-282. Unit Masks for RxR_BYPASS

Extension	umask [15:8]	xtra [57:32]	Description
AD_UNCRD	b00000001	bx1xxxxxx xxxxxxxxx xxxxxxxx	AD - Uncredited
AK	b00000010	bx1xxxxxx xxxxxxxxx xxxxxxxx	AK
BL_UNCRD	b00000100	bx1xxxxxx xxxxxxxxx xxxxxxxx	BL - Uncredited
IV	b00001000	bx1xxxxxx xxxxxxxxx xxxxxxxx	IV
AD_CRD	b00010000	bx1xxxxxx xxxxxxxxx xxxxxxxx	AD - Credited
AD_ALL	b00010001	bx1xxxxxx xxxxxxxxx xxxxxxxx	AD - All All == Credited + Uncredited
BL_CRD	b01000000	bx1xxxxxx xxxxxxxxx xxxxxxxx	BL - Credited
BL_ALL	b01000100	bx1xxxxxx xxxxxxxxx xxxxxxxx	BL - All All == Credited + Uncredited
AKC_UNCRD	b10000000	bx1xxxxxx xxxxxxxxx xxxxxxxx	AKC - Uncredited

RxR_CRD_STARVED

- **Title:**
- **Category:** Ingress Transgress Events
- **Event Code:** 0xE3
- **Register Restrictions :**
- **Definition:** Counts cycles under injection starvation mode. This starvation is triggered when the CMS ingress cannot send a transaction onto the mesh for a long period of time. In this case, the ingress is unable to forward to the egress due to a lack of credit.

Table 2-283. Unit Masks for RxR_CRD_STARVED (Sheet 1 of 2)

Extension	umask [15:8]	xtra [57:32]	Description
AD_UNCRD	b00000001	bx1xxxxxx xxxxxxxxx xxxxxxxx	AD - Uncredited
AK	b00000010	bx1xxxxxx xxxxxxxxx xxxxxxxx	AK
BL_UNCRD	b00000100	bx1xxxxxx xxxxxxxxx xxxxxxxx	BL - Uncredited

Table 2-283. Unit Masks for RxR_CRD_STARVED (Sheet 2 of 2)

Extension	umask [15:8]	xtra [57:32]	Description
IV	b00001000	bx1xxxxxx xxxxxxxxx xxxxxxxx	IV
AD_CRD	b00010000	bx1xxxxxx xxxxxxxxx xxxxxxxx	AD - Credited
AD_ALL	b00010001	bx1xxxxxx xxxxxxxxx xxxxxxxx	AD - All All == Credited + Uncredited
BL_CRD	b01000000	bx1xxxxxx xxxxxxxxx xxxxxxxx	BL - Credited
BL_ALL	b01000100	bx1xxxxxx xxxxxxxxx xxxxxxxx	BL - All All == Credited + Uncredited
IFV	b10000000	bx1xxxxxx xxxxxxxxx xxxxxxxx	IFV - Credited

RxR_CRD_STARVED_1

- **Title:**
- **Category:** Ingress Transgress Events
- **Event Code:** 0xE4
- **Register Restrictions :**
- **Definition:** Counts cycles under injection starvation mode. This starvation is triggered when the CMS ingress cannot send a transaction onto the mesh for a long period of time. In this case, the ingress is unable to forward to the egress due to a lack of credit.

Table 2-284. Unit Masks for RxR_CRD_STARVED_1

Extension	umask [15:8]	xtra [57:32]	Description
AKC_UNCRD	b10000000	bx1xxxxxx xxxxxxxxx xxxxxxxx	AKC - Uncredited

RxR_INSERTS

- **Title:**
- **Category:** Ingress Transgress Events
- **Event Code:** 0xE1
- **Register Restrictions :**
- **Definition:** Number of allocations into the CMS ingress. The ingress is used to queue up requests received from the mesh.

Table 2-285. Unit Masks for RxR_INSERTS

Extension	umask [15:8]	xtra [57:32]	Description
AD_UNCRD	b00000001	bx1xxxxxx xxxxxxxx xxxxxxxx	AD - Uncredited
AK	b00000010	bx1xxxxxx xxxxxxxx xxxxxxxx	AK
BL_UNCRD	b00000100	bx1xxxxxx xxxxxxxx xxxxxxxx	BL - Uncredited
IV	b00001000	bx1xxxxxx xxxxxxxx xxxxxxxx	IV
AD_CRD	b00010000	bx1xxxxxx xxxxxxxx xxxxxxxx	AD - Credited
AD_ALL	b00010001	bx1xxxxxx xxxxxxxx xxxxxxxx	AD - All All == Credited + Uncredited
BL_CRD	b01000000	bx1xxxxxx xxxxxxxx xxxxxxxx	BL - Credited
BL_ALL	b01000100	bx1xxxxxx xxxxxxxx xxxxxxxx	BL - All All == Credited + Uncredited
AKC_UNCRD	b10000000	bx1xxxxxx xxxxxxxx xxxxxxxx	AKC - Uncredited

RxR_OCCUPANCY

- **Title:**
- **Category:** Ingress Transgress Events
- **Event Code:** 0xE0
- **Register Restrictions :**
- **Definition:** Occupancy event for the ingress buffers in the CMS. The ingress is used to queue up requests received from the mesh.

Table 2-286. Unit Masks for RxR_OCCUPANCY (Sheet 1 of 2)

Extension	umask [15:8]	xtra [57:32]	Description
AD_UNCRD	b00000001	bx1xxxxxx xxxxxxxx xxxxxxxx	AD - Uncredited
AK	b00000010	bx1xxxxxx xxxxxxxx xxxxxxxx	AK
BL_UNCRD	b00000100	bx1xxxxxx xxxxxxxx xxxxxxxx	BL - Uncredited
IV	b00001000	bx1xxxxxx xxxxxxxx xxxxxxxx	IV

Table 2-286. Unit Masks for RxR_OCCUPANCY (Sheet 2 of 2)

Extension	umask [15:8]	xtra [57:32]	Description
AD_CRD	b00010000	bx1xxxxxx xxxxxxxxx xxxxxxxx	AD - Credited
AD_ALL	b00010001	bx1xxxxxx xxxxxxxxx xxxxxxxx	AD - All All == Credited + Uncredited
BL_CRD	b00100000	bx1xxxxxx xxxxxxxxx xxxxxxxx	BL - Credited
BL_ALL	b01000100	bx1xxxxxx xxxxxxxxx xxxxxxxx	BL - All All == Credited + Uncredited
AKC_UNCRD	b10000000	bx1xxxxxx xxxxxxxxx xxxxxxxx	AKC - Uncredited

STALL0_NO_TxR_HORZ_CRD_AD_AG0

- **Title:**
- **Category:** Transgress Credit Events
- **Event Code:** 0xD0
- **Register Restrictions :**
- **Definition:** Number of cycles the AD agent 0 egress buffer is stalled waiting for a TGR credit to become available, per transgress.

Table 2-287. Unit Masks for STALL0_NO_TxR_HORZ_CRD_AD_AG0

Extension	umask [15:8]	xtra [57:32]	Description
TGR0	bxxxxxxx1	bx1xxxxxx xxxxxxxxx xxxxxxxx	For Transgress 0
TGR1	bxxxxxx1x	bx1xxxxxx xxxxxxxxx xxxxxxxx	For Transgress 1
TGR2	bxxxxx1xx	bx1xxxxxx xxxxxxxxx xxxxxxxx	For Transgress 2
TGR3	bxxxx1xxx	bx1xxxxxx xxxxxxxxx xxxxxxxx	For Transgress 3
TGR4	bxxx1xxxx	bx1xxxxxx xxxxxxxxx xxxxxxxx	For Transgress 4
TGR5	bxx1xxxxx	bx1xxxxxx xxxxxxxxx xxxxxxxx	For Transgress 5
TGR6	bx1xxxxxx	bx1xxxxxx xxxxxxxxx xxxxxxxx	For Transgress 6
TGR7	b1xxxxxxx	bx1xxxxxx xxxxxxxxx xxxxxxxx	For Transgress 7

STALL0_NO_TxR_HORZ_CRD_AD_AG1

- **Title:**
- **Category:** Transgress Credit Events
- **Event Code:** 0xD2
- **Register Restrictions :**
- **Definition:** Number of cycles the AD agent 1 egress buffer is stalled waiting for a TGR credit to become available, per transgress.

Table 2-288. Unit Masks for STALL0_NO_TxR_HORZ_CRD_AD_AG1

Extension	umask [15:8]	xtra [57:32]	Description
TGR0	bxxxxxxx1	bx1xxxxxx xxxxxxxxx xxxxxxxx	For Transgress 0
TGR1	bxxxxxx1x	bx1xxxxxx xxxxxxxxx xxxxxxxx	For Transgress 1
TGR2	bxxxxx1xx	bx1xxxxxx xxxxxxxxx xxxxxxxx	For Transgress 2
TGR3	bxxxx1xxx	bx1xxxxxx xxxxxxxxx xxxxxxxx	For Transgress 3
TGR4	bxxx1xxxx	bx1xxxxxx xxxxxxxxx xxxxxxxx	For Transgress 4
TGR5	bxx1xxxxx	bx1xxxxxx xxxxxxxxx xxxxxxxx	For Transgress 5
TGR6	bx1xxxxxx	bx1xxxxxx xxxxxxxxx xxxxxxxx	For Transgress 6
TGR7	b1xxxxxxx	bx1xxxxxx xxxxxxxxx xxxxxxxx	For Transgress 7

STALL0_NO_TxR_HORZ_CRD_BL_AG0

- **Title:**
- **Category:** Transgress Credit Events
- **Event Code:** 0xD4
- **Register Restrictions :**
- **Definition:** Number of cycles the BL agent 0 egress buffer is stalled waiting for a TGR credit to become available, per transgress.

Table 2-289. Unit Masks for STALL0_NO_TxR_HORZ_CRD_BL_AG0 (Sheet 1 of 2)

Extension	umask [15:8]	xtra [57:32]	Description
TGR0	bxxxxxxx1	bx1xxxxxx xxxxxxxxx xxxxxxxx	For Transgress 0
TGR1	bxxxxxx1x	bx1xxxxxx xxxxxxxxx xxxxxxxx	For Transgress 1

Table 2-289. Unit Masks for STALL0_NO_TxR_HORZ_CRD_BL_AG0 (Sheet 2 of 2)

Extension	umask [15:8]	xtra [57:32]	Description
TGR2	bxxxxx1xx	bx1xxxxxx xxxxxxxxx xxxxxxxx	For Transgress 2
TGR3	bxxxx1xxx	bx1xxxxxx xxxxxxxxx xxxxxxxx	For Transgress 3
TGR4	bxxx1xxxx	bx1xxxxxx xxxxxxxxx xxxxxxxx	For Transgress 4
TGR5	bxx1xxxxx	bx1xxxxxx xxxxxxxxx xxxxxxxx	For Transgress 5
TGR6	bx1xxxxxx	bx1xxxxxx xxxxxxxxx xxxxxxxx	For Transgress 6
TGR7	b1xxxxxxx	bx1xxxxxx xxxxxxxxx xxxxxxxx	For Transgress 7

STALL0_NO_TxR_HORZ_CRD_BL_AG1

- **Title:**
- **Category:** Transgress Credit Events
- **Event Code:** 0xD6
- **Register Restrictions :**
- **Definition:** Number of cycles the BL agent 1 egress buffer is stalled waiting for a TGR credit to become available, per transgress.

Table 2-290. Unit Masks for STALL0_NO_TxR_HORZ_CRD_BL_AG1 (Sheet 1 of 2)

Extension	umask [15:8]	xtra [57:32]	Description
TGR0	bxxxxxxx1	bx1xxxxxx xxxxxxxxx xxxxxxxx	For Transgress 0
TGR1	bxxxxxx1x	bx1xxxxxx xxxxxxxxx xxxxxxxx	For Transgress 1
TGR2	bxxxxx1xx	bx1xxxxxx xxxxxxxxx xxxxxxxx	For Transgress 2
TGR3	bxxxx1xxx	bx1xxxxxx xxxxxxxxx xxxxxxxx	For Transgress 3
TGR4	bxxx1xxxx	bx1xxxxxx xxxxxxxxx xxxxxxxx	For Transgress 4
TGR5	bxx1xxxxx	bx1xxxxxx xxxxxxxxx xxxxxxxx	For Transgress 5

Table 2-290. Unit Masks for STALL0_NO_TxR_HORZ_CRD_BL_AG1 (Sheet 2 of 2)

Extension	umask [15:8]	xtra [57:32]	Description
TGR6	bx1xxxxxx	bx1xxxxxx xxxxxxxxx xxxxxxxx	For Transgress 6
TGR7	b1xxxxxxx	bx1xxxxxx xxxxxxxxx xxxxxxxx	For Transgress 7

STALL1_NO_TxR_HORZ_CRD_AD_AG0

- **Title:**
- **Category:** Transgress Credit Events
- **Event Code:** 0xD1
- **Register Restrictions :**
- **Definition:** Number of cycles the AD agent 0 egress buffer is stalled waiting for a TGR credit to become available, per transgress.

Table 2-291. Unit Masks for STALL1_NO_TxR_HORZ_CRD_AD_AG0

Extension	umask [15:8]	xtra [57:32]	Description
TGR8	bxxxxxxx1	bx1xxxxxx xxxxxxxxx xxxxxxxx	For Transgress 8
TGR9	bxxxxxx1x	bx1xxxxxx xxxxxxxxx xxxxxxxx	For Transgress 9
TGR10	bxxxxx1xx	bx1xxxxxx xxxxxxxxx xxxxxxxx	For Transgress 10
TGR11	bxxxx1xxx	bx1xxxxxx xxxxxxxxx xxxxxxxx	For Transgress 11
TGR12	bxxx1xxxx	bx1xxxxxx xxxxxxxxx xxxxxxxx	For Transgress 12
TGR13	bxx1xxxxx	bx1xxxxxx xxxxxxxxx xxxxxxxx	For Transgress 13
TGR14	bx1xxxxxx	bx1xxxxxx xxxxxxxxx xxxxxxxx	For Transgress 14
TGR15	b1xxxxxxx	bx1xxxxxx xxxxxxxxx xxxxxxxx	For Transgress 15

STALL1_NO_TxR_HORZ_CRD_AD_AG1

- **Title:**
- **Category:** Transgress Credit Events
- **Event Code:** 0xD3
- **Register Restrictions :**
- **Definition:** Number of cycles the AD agent 1 egress buffer is stalled waiting for a TGR credit to become available, per transgress.

Table 2-292. Unit Masks for STALL1_NO_TxR_HORZ_CRD_AD_AG1

Extension	umask [15:8]	xtra [57:32]	Description
TGR8	bxxxxxxx1	bx1xxxxxx xxxxxxxxx xxxxxxxx	For Transgress 8
TGR9	bxxxxxx1x	bx1xxxxxx xxxxxxxxx xxxxxxxx	For Transgress 9
TGR10	bxxxxx1xx	bx1xxxxxx xxxxxxxxx xxxxxxxx	For Transgress 10
TGR11	bxxxx1xxx	bx1xxxxxx xxxxxxxxx xxxxxxxx	For Transgress 11
TGR12	bxxx1xxxx	bx1xxxxxx xxxxxxxxx xxxxxxxx	For Transgress 12
TGR13	bxx1xxxxx	bx1xxxxxx xxxxxxxxx xxxxxxxx	For Transgress 13
TGR14	bx1xxxxxx	bx1xxxxxx xxxxxxxxx xxxxxxxx	For Transgress 14
TGR15	b1xxxxxxx	bx1xxxxxx xxxxxxxxx xxxxxxxx	For Transgress 15

STALL1_NO_TxR_HORZ_CRD_BL_AG0

- **Title:**
- **Category:** Transgress Credit Events
- **Event Code:** 0xD5
- **Register Restrictions :**
- **Definition:** Number of cycles the BL agent 0 egress buffer is stalled waiting for a TGR credit to become available, per transgress.

Table 2-293. Unit Masks for STALL1_NO_TxR_HORZ_CRD_BL_AG0 (Sheet 1 of 2)

Extension	umask [15:8]	xtra [57:32]	Description
TGR8	bxxxxxxx1	bx1xxxxxx xxxxxxxxx xxxxxxxx	For Transgress 8
TGR9	bxxxxxx1x	bx1xxxxxx xxxxxxxxx xxxxxxxx	For Transgress 9
TGR10	bxxxxx1xx	bx1xxxxxx xxxxxxxxx xxxxxxxx	For Transgress 10
TGR11	bxxxx1xxx	bx1xxxxxx xxxxxxxxx xxxxxxxx	For Transgress 11
TGR12	bxxx1xxxx	bx1xxxxxx xxxxxxxxx xxxxxxxx	For Transgress 12

Table 2-293. Unit Masks for STALL1_NO_TxR_HORZ_CRD_BL_AG0 (Sheet 2 of 2)

Extension	umask [15:8]	xtra [57:32]	Description
TGR13	bxx1xxxxx	bx1xxxxxx xxxxxxxxx xxxxxxxx	For Transgress 13
TGR14	bx1xxxxxx	bx1xxxxxx xxxxxxxxx xxxxxxxx	For Transgress 14
TGR15	b1xxxxxxx	bx1xxxxxx xxxxxxxxx xxxxxxxx	For Transgress 15

STALL1_NO_TxR_HORZ_CRD_BL_AG1

- **Title:**
- **Category:** Transgress Credit Events
- **Event Code:** 0xD7
- **Register Restrictions :**
- **Definition:** Number of cycles the BL agent 1 egress buffer is stalled waiting for a TGR credit to become available, per transgress.

Table 2-294. Unit Masks for STALL1_NO_TxR_HORZ_CRD_BL_AG1

Extension	umask [15:8]	xtra [57:32]	Description
TGR8	bxxxxxxx1	bx1xxxxxx xxxxxxxxx xxxxxxxx	For Transgress 8
TGR9	bxxxxxx1x	bx1xxxxxx xxxxxxxxx xxxxxxxx	For Transgress 9
TGR10	bxxxxx1xx	bx1xxxxxx xxxxxxxxx xxxxxxxx	For Transgress 10
TGR11	bxxxx1xxx	bx1xxxxxx xxxxxxxxx xxxxxxxx	For Transgress 11
TGR12	bxxx1xxxx	bx1xxxxxx xxxxxxxxx xxxxxxxx	For Transgress 12
TGR13	bxx1xxxxx	bx1xxxxxx xxxxxxxxx xxxxxxxx	For Transgress 13
TGR14	bx1xxxxxx	bx1xxxxxx xxxxxxxxx xxxxxxxx	For Transgress 14
TGR15	b1xxxxxxx	bx1xxxxxx xxxxxxxxx xxxxxxxx	For Transgress 15

TAG_HIT

- **Title:**
- **Category:** Directory State Events
- **Event Code:** 0x1F
- **Register Restrictions :** 0-3

- **Definition:** Tag hit indicates when a request sent to the iMC hit in the near memory.

Table 2-295. Unit Masks for TAG_HIT

Extension	umask [15:8]	Description
NM_RD_HIT_CLEAN	bxxxxxxx1	Clean NearMem Read Hit Counts clean full line read hits (reads and RFOs)
NM_RD_HIT_DIRTY	bxxxxxx1x	Dirty NearMem Read Hit Counts dirty full line read hits (reads and RFOs)
NM_UFILL_HIT_CLEAN	bxxxxx1xx	Clean NearMem Underfill Hit Counts clean underfill hits due to a partial write
NM_UFILL_HIT_DIRTY	bxxxx1xxx	Dirty NearMem Underfill Hit Counts dirty underfill read hits due to a partial write

TAG_MISS

- **Title:**
- **Category:** Directory State Events
- **Event Code:** 0x4B
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-296. Unit Masks for TAG_MISS

Extension	umask [15:8]	Description
PMM	bxxxxxxx1	Counts the 2lm miss case for any data return
PMM_TWO_WAY	bxxxxxx1x	Counts the 2lm miss case by qualifying with 2way. For 2way, we only generate a fill when we miss both ways

TGR_AD_CREDITS

- **Title:**
- **Category:** Transgress Credit Events
- **Event Code:** 0x2E
- **Register Restrictions :** 0-3
- **Definition:**

TGR_BL_CREDITS

- **Title:**
- **Category:** Transgress Credit Events
- **Event Code:** 0x2F
- **Register Restrictions :** 0-3
- **Definition:**

TxC_AD

- **Title:**
- **Category:** AD Egress Events
- **Event Code:** 0x6
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-297. Unit Masks for TxC_AD

Extension	umask [15:8]	Description
INSERTS	bxxxxxxx1	AD Egress (to CMS) Allocations Counts anytime a AD packet is added to egress
CYCLES_NE	bxxxxxx1x	AD Egress (to CMS) Not Empty Counts the number of cycles AD egress is not empty
CYCLES_FULL	bxxxxx1xx	AD Egress (to CMS) Full Counts the number of cycles AD egress is full

TxC_BL_CREDITS

- **Title:**
- **Category:** BL CMS/Mesh Egress Credit Events
- **Event Code:** 0x10
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-298. Unit Masks for TxC_BL_CREDITS

Extension	umask [15:8]	xtra [57:32]	Description
RETURNED_CMS0	bxxxxxxx1	bxxxxxxx1	Credit Returns - CMS0 Counts the number of credit returns for ADD 0 port
RETURNED_CMS1	bxxxxxxx1	bxxxxxx1x	Credit Returns - CMS1 Counts the number of credit returns for ADD 1 port
ZERO_CMS0	bxxxxxx1x	bxxxxxxx1	Zero Credits - CMS0 Counts the number of cycles where the ADD 0 credit is zero
ZERO_CMS1	bxxxxxx1x	bxxxxxx1x	Zero Credits - CMS1 Counts the number of cycles where the ADD 1 credit is zero
STALLED_CMS0	bxxxxx1xx	bxxxxxxx1	Stalled, No Credits - CMS0 Counts the number of cycles for which BL egress is stalled due to no credit on add 0
STALLED_CMS1	bxxxxx1xx	bxxxxxx1x	Stalled, No Credits - CMS1 Counts the number of cycles for which BL egress is stalled due to no credit on add 1

TxC_EGRESS_AK

- **Title:**
- **Category:** AK Egress Events
- **Event Code:** 0xA
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-299. Unit Masks for TxC_EGRESS_AK

Extension	umask [15:8]	Description
INSERTS	bxxxxxxx1	Allocations
CYCLES_NE	bxxxxxx1x	Not Empty
CYCLES_FULL	bxxxxx1xx	Full

TxR_HORZ_ADS_USED

- **Title:**
- **Category:** Horizontal Egress Events
- **Event Code:** 0xA6
- **Register Restrictions :**
- **Definition:** Number of packets using the horizontal anti-deadlock slot, broken down by ring type and CMS agent.

Table 2-300. Unit Masks for TxR_HORZ_ADS_USED

Extension	umask [15:8]	xtra [57:32]	Description
AD_UNCRD	b00000001	bx1xxxxxx xxxxxxxxx xxxxxxxx	AD - Uncredited
BL_UNCRD	b00000100	bx1xxxxxx xxxxxxxxx xxxxxxxx	BL - Uncredited
AD_CRD	b00010000	bx1xxxxxx xxxxxxxxx xxxxxxxx	AD - Credited
AD_ALL	b00010001	bx1xxxxxx xxxxxxxxx xxxxxxxx	AD - All All == Credited + Uncredited
BL_CRD	b01000000	bx1xxxxxx xxxxxxxxx xxxxxxxx	BL - Credited
BL_ALL	b01000100	bx1xxxxxx xxxxxxxxx xxxxxxxx	BL - All All == Credited + Uncredited

TxR_HORZ_BYPASS

- **Title:**
- **Category:** Horizontal Egress Events
- **Event Code:** 0xA7
- **Register Restrictions :**
- **Definition:** Number of packets bypassing the horizontal egress, broken down by ring type and CMS agent.

Table 2-301. Unit Masks for TxR_HORZ_BYPASS

Extension	umask [15:8]	xtra [57:32]	Description
AD_UNCRD	b00000001	bx1xxxxxxx xxxxxxxxxx xxxxxx	AD - Uncredited
AK	b00000010	bx1xxxxxxx xxxxxxxxxx xxxxxx	AK
BL_UNCRD	b00000100	bx1xxxxxxx xxxxxxxxxx xxxxxx	BL - Uncredited
IV	b00001000	bx1xxxxxxx xxxxxxxxxx xxxxxx	IV
AD_CRD	b00010000	bx1xxxxxxx xxxxxxxxxx xxxxxx	AD - Credited
AD_ALL	b00010001	bx1xxxxxxx xxxxxxxxxx xxxxxx	AD - All All == Credited + Uncredited
BL_CRD	b01000000	bx1xxxxxxx xxxxxxxxxx xxxxxx	BL - Credited
BL_ALL	b01000100	bx1xxxxxxx xxxxxxxxxx xxxxxx	BL - All All == Credited + Uncredited
AKC_UNCRD	b10000000	bx1xxxxxxx xxxxxxxxxx xxxxxx	AKC - Uncredited

TxR_HORZ_CYCLES_FULL

- **Title:**
- **Category:** Horizontal Egress Events
- **Event Code:** 0xA2
- **Register Restrictions :**
- **Definition:** Cycles the transgress buffers in the common mesh stop are full. The egress is used to queue up requests destined for the horizontal ring on the mesh.

Table 2-302. Unit Masks for TxR_HORZ_CYCLES_FULL (Sheet 1 of 2)

Extension	umask [15:8]	xtra [57:32]	Description
AD_UNCRD	b00000001	bx1xxxxxxx xxxxxxxxxx xxxxxx	AD - Uncredited
AK	b00000010	bx1xxxxxxx xxxxxxxxxx xxxxxx	AK
BL_UNCRD	b00000100	bx1xxxxxxx xxxxxxxxxx xxxxxx	BL - Uncredited
IV	b00001000	bx1xxxxxxx xxxxxxxxxx xxxxxx	IV

Table 2-302. Unit Masks for TxR_HORZ_CYCLES_FULL (Sheet 2 of 2)

Extension	umask [15:8]	xtra [57:32]	Description
AD_CRD	b00010000	bx1xxxxxxx xxxxxxxxxx xxxxxx	AD - Credited
AD_ALL	b00010001	bx1xxxxxxx xxxxxxxxxx xxxxxx	AD - All All == Credited + Uncredited
BL_CRD	b01000000	bx1xxxxxxx xxxxxxxxxx xxxxxx	BL - Credited
BL_ALL	b01000100	bx1xxxxxxx xxxxxxxxxx xxxxxx	BL - All All == Credited + Uncredited
AKC_UNCRD	b10000000	bx1xxxxxxx xxxxxxxxxx xxxxxx	AKC - Uncredited

TxR_HORZ_CYCLES_NE

- **Title:**
- **Category:** Horizontal Egress Events
- **Event Code:** 0xA3
- **Register Restrictions :**
- **Definition:** Cycles the transgress buffers in the common mesh stop are not-empty. The egress is used to queue up requests destined for the horizontal ring on the mesh.

Table 2-303. Unit Masks for TxR_HORZ_CYCLES_NE (Sheet 1 of 2)

Extension	umask [15:8]	xtra [57:32]	Description
AD_UNCRD	b00000001	bx1xxxxxxx xxxxxxxxxx xxxxxx	AD - Uncredited
AK	b00000010	bx1xxxxxxx xxxxxxxxxx xxxxxx	AK
BL_UNCRD	b00000100	bx1xxxxxxx xxxxxxxxxx xxxxxx	BL - Uncredited
IV	b00001000	bx1xxxxxxx xxxxxxxxxx xxxxxx	IV
AD_CRD	b00010000	bx1xxxxxxx xxxxxxxxxx xxxxxx	AD - Credited
AD_ALL	b00010001	bx1xxxxxxx xxxxxxxxxx xxxxxx	AD - All All == Credited + Uncredited
BL_CRD	b01000000	bx1xxxxxxx xxxxxxxxxx xxxxxx	BL - Credited

Table 2-303. Unit Masks for TxR_HORZ_CYCLES_NE (Sheet 2 of 2)

Extension	umask [15:8]	xtra [57:32]	Description
BL_ALL	b01000100	bx1xxxxxxx xxxxxxxxx xxxxxx	BL - All All == Credited + Uncredited
AKC_UNCRD	b10000000	bx1xxxxxxx xxxxxxxxx xxxxxx	AKC - Uncredited

TxR_HORZ_INSERTS

- **Title:**
- **Category:** Horizontal Egress Events
- **Event Code:** 0xA1
- **Register Restrictions :**
- **Definition:** Number of allocations into the transgress buffers in the common mesh stop. The egress is used to queue up requests destined for the horizontal ring on the mesh.

Table 2-304. Unit Masks for TxR_HORZ_INSERTS

Extension	umask [15:8]	xtra [57:32]	Description
AD_UNCRD	b00000001	bx1xxxxxxx xxxxxxxxx xxxxxx	AD - Uncredited
AK	b00000010	bx1xxxxxxx xxxxxxxxx xxxxxx	AK
BL_UNCRD	b00000100	bx1xxxxxxx xxxxxxxxx xxxxxx	BL - Uncredited
IV	b00001000	bx1xxxxxxx xxxxxxxxx xxxxxx	IV
AD_CRD	b00010000	bx1xxxxxxx xxxxxxxxx xxxxxx	AD - Credited
AD_ALL	b00010001	bx1xxxxxxx xxxxxxxxx xxxxxx	AD - All All == Credited + Uncredited
BL_CRD	b01000000	bx1xxxxxxx xxxxxxxxx xxxxxx	BL - Credited
BL_ALL	b01000100	bx1xxxxxxx xxxxxxxxx xxxxxx	BL - All All == Credited + Uncredited
AKC_UNCRD	b10000000	bx1xxxxxxx xxxxxxxxx xxxxxx	AKC - Uncredited

TxR_HORZ_NACK

- **Title:**
- **Category:** Horizontal Egress Events
- **Event Code:** 0xA4
- **Register Restrictions :**
- **Definition:** Counts number of egress packets NACKed on to the horizontal ring.

Table 2-305. Unit Masks for TxR_HORZ_NACK

Extension	umask [15:8]	xtra [57:32]	Description
AD_UNCRD	b00000001	bx1xxxxxxx xxxxxxxxxx xxxxxx	AD - Uncredited
AK	b00000010	bx1xxxxxxx xxxxxxxxxx xxxxxx	AK
BL_UNCRD	b00000100	bx1xxxxxxx xxxxxxxxxx xxxxxx	BL - Uncredited
IV	b00001000	bx1xxxxxxx xxxxxxxxxx xxxxxx	IV
AD_CRD	b00010000	bx1xxxxxxx xxxxxxxxxx xxxxxx	AD - Credited
AD_ALL	b00010001	bx1xxxxxxx xxxxxxxxxx xxxxxx	AD - All All == Credited + Uncredited
BL_CRD	b01000000	bx1xxxxxxx xxxxxxxxxx xxxxxx	BL - Credited
BL_ALL	b01000100	bx1xxxxxxx xxxxxxxxxx xxxxxx	BL - All All == Credited + Uncredited
AKC_UNCRD	b10000000	bx1xxxxxxx xxxxxxxxxx xxxxxx	AKC - Uncredited

TxR_HORZ_OCCUPANCY

- **Title:**
- **Category:** Horizontal Egress Events
- **Event Code:** 0xA0
- **Register Restrictions :**
- **Definition:** Occupancy event for the transgress buffers in the common mesh stop. The egress is used to queue up requests destined for the horizontal ring on the mesh.

Table 2-306. Unit Masks for TxR_HORZ_OCCUPANCY

Extension	umask [15:8]	xtra [57:32]	Description
AD_UNCRD	b00000001	bx1xxxxxxxxxxxxxxxxxxxxxxx	AD - Uncredited
AK	b00000010	bx1xxxxxxxxxxxxxxxxxxxxxxx	AK
BL_UNCRD	b00000100	bx1xxxxxxxxxxxxxxxxxxxxxxx	BL - Uncredited
IV	b00001000	bx1xxxxxxxxxxxxxxxxxxxxxxx	IV
AD_CRD	b00010000	bx1xxxxxxxxxxxxxxxxxxxxxxx	AD - Credited
AD_ALL	b00010001	bx1xxxxxxxxxxxxxxxxxxxxxxx	AD - All All == Credited + Uncredited
BL_CRD	b01000000	bx1xxxxxxxxxxxxxxxxxxxxxxx	BL - Credited
BL_ALL	b01000100	bx1xxxxxxxxxxxxxxxxxxxxxxx	BL - All All == Credited + Uncredited
AKC_UNCRD	b10000000	bx1xxxxxxxxxxxxxxxxxxxxxxx	AKC - Uncredited

TxR_HORZ_STARVED

- **Title:**
- **Category:** Horizontal Egress Events
- **Event Code:** 0xA5
- **Register Restrictions :**
- **Definition:** Counts injection starvation. This starvation is triggered when the CMS transgress buffer cannot send a transaction onto the horizontal ring for a long period of time.

Table 2-307. Unit Masks for TxR_HORZ_STARVED (Sheet 1 of 2)

Extension	umask [15:8]	xtra [57:32]	Description
AD_ALL	b00000001	bx1xxxxxxxxxxxxxxxxxxxxxxx	AD - All All == Credited + Uncredited
AD_UNCRD	b00000001	bx1xxxxxxxxxxxxxxxxxxxxxxx	AD - Uncredited
AK	b00000010	bx1xxxxxxxxxxxxxxxxxxxxxxx	AK

Table 2-307. Unit Masks for TxR_HORZ_STARVED (Sheet 2 of 2)

Extension	umask [15:8]	xtra [57:32]	Description
BL_ALL	b00000100	bx1xxxxxxx xxxxxxxxxx xxxxxx	BL - All All == Credited + Uncredited
BL_UNCRD	b00000100	bx1xxxxxxx xxxxxxxxxx xxxxxx	BL - Uncredited
IV	b00001000	bx1xxxxxxx xxxxxxxxxx xxxxxx	IV
AKC_UNCRD	b10000000	bx1xxxxxxx xxxxxxxxxx xxxxxx	AKC - Uncredited

TxR_VERT_ADS_USED

- **Title:**
- **Category:** Vertical Egress Events
- **Event Code:** 0x9C
- **Register Restrictions :**
- **Definition:** Number of packets using the vertical anti-deadlock slot, broken down by ring type and CMS agent.

Table 2-308. Unit Masks for TxR_VERT_ADS_USED

Extension	umask [15:8]	xtra [57:32]	Description
AD_AG0	bxxxxxxx1	bx1xxxxxxx xxxxxxxxxx xxxxxx	AD - Agent 0
BL_AG0	bxxxxx1xx	bx1xxxxxxx xxxxxxxxxx xxxxxx	BL - Agent 0
AD_AG1	bxxx1xxxx	bx1xxxxxxx xxxxxxxxxx xxxxxx	AD - Agent 1
BL_AG1	bx1xxxxxx	bx1xxxxxxx xxxxxxxxxx xxxxxx	BL - Agent 1

TxR_VERT_BYPASS

- **Title:**
- **Category:** Vertical Egress Events
- **Event Code:** 0x9D
- **Register Restrictions :**
- **Definition:** Number of packets bypassing the vertical egress, broken down by ring type and CMS agent.

Table 2-309. Unit Masks for TxR_VERT_BYPASS

Extension	umask [15:8]	xtra [57:32]	Description
AD_AG0	bxxxxxxx1	bx1xxxxxxx xxxxxxxxxx xxxxxx	AD - Agent 0
AK_AG0	bxxxxxx1x	bx1xxxxxxx xxxxxxxxxx xxxxxx	AK - Agent 0
BL_AG0	bxxxxx1xx	bx1xxxxxxx xxxxxxxxxx xxxxxx	BL - Agent 0
IV_AG1	bxxxx1xxx	bx1xxxxxxx xxxxxxxxxx xxxxxx	IV - Agent 1
AD_AG1	bxxx1xxxx	bx1xxxxxxx xxxxxxxxxx xxxxxx	AD - Agent 1
AK_AG1	bxx1xxxxx	bx1xxxxxxx xxxxxxxxxx xxxxxx	AK - Agent 1
BL_AG1	bx1xxxxxx	bx1xxxxxxx xxxxxxxxxx xxxxxx	BL - Agent 1

TxR_VERT_BYPASS_1

- **Title:**
- **Category:** Vertical Egress Events
- **Event Code:** 0x9E
- **Register Restrictions :**
- **Definition:** Number of packets bypassing the vertical egress, broken down by ring type and CMS agent.

Table 2-310. Unit Masks for TxR_VERT_BYPASS_1

Extension	umask [15:8]	xtra [57:32]	Description
AKC_AG0	bxxxxxxx1	bx1xxxxxxx xxxxxxxxxx xxxxxx	AKC - Agent 0
AKC_AG1	bxxxxxx1x	bx1xxxxxxx xxxxxxxxxx xxxxxx	AKC - Agent 1

TxR_VERT_CYCLES_FULL0

- **Title:**
- **Category:** Vertical Egress Events
- **Event Code:** 0x94
- **Register Restrictions :**
- **Definition:** Number of cycles the common mesh stop egress was not full. The egress is used to queue up requests destined for the vertical ring on the mesh.

Table 2-311. Unit Masks for TxR_VERT_CYCLES_FULL0

Extension	umask [15:8]	xtra [57:32]	Description
AD_AG0	bxxxxxxx1	bx1xxxxxxx xxxxxxxxxx xxxxxx	AD - Agent 0 Ring transactions from Agent 0 destined for the AD ring. Some example include outbound requests, snoop requests, and snoop responses.
AK_AG0	bxxxxxx1x	bx1xxxxxxx xxxxxxxxxx xxxxxx	AK - Agent 0 Ring transactions from Agent 0 destined for the AK ring. This is commonly used for credit returns and GO responses.
BL_AG0	bxxxxx1xx	bx1xxxxxxx xxxxxxxxxx xxxxxx	BL - Agent 0 Ring transactions from Agent 0 destined for the BL ring. This is commonly used to send data from the cache to various destinations.
IV_AG0	bxxxx1xxx	bx1xxxxxxx xxxxxxxxxx xxxxxx	IV - Agent 0 Ring transactions from Agent 0 destined for the IV ring. This is commonly used for snoops to the cores.
AD_AG1	bxxx1xxxx	bx1xxxxxxx xxxxxxxxxx xxxxxx	AD - Agent 1 Ring transactions from Agent 1 destined for the AD ring. This is commonly used for outbound requests.
AK_AG1	bxx1xxxxx	bx1xxxxxxx xxxxxxxxxx xxxxxx	AK - Agent 1 Ring transactions from Agent 1 destined for the AK ring.
BL_AG1	bx1xxxxxx	bx1xxxxxxx xxxxxxxxxx xxxxxx	BL - Agent 1 Ring transactions from Agent 1 destined for the BL ring. This is commonly used for transferring write back data to the cache.

TxR_VERT_CYCLES_FULL1

- **Title:**
- **Category:** Vertical Egress Events
- **Event Code:** 0x95
- **Register Restrictions :**
- **Definition:** Number of cycles the common mesh stop egress was not full. The egress is used to queue up requests destined for the vertical ring on the mesh.

Table 2-312. Unit Masks for TxR_VERT_CYCLES_FULL1

Extension	umask [15:8]	xtra [57:32]	Description
AKC_AG0	bxxxxxxx1	bx1xxxxxxx xxxxxxxxxx xxxxxx	AKC - Agent 0 Ring transactions from Agent 0 destined for the AD ring. Some example include outbound requests, snoop requests, and snoop responses.
AKC_AG1	bxxxxxx1x	bx1xxxxxxx xxxxxxxxxx xxxxxx	AKC - Agent 1 Ring transactions from Agent 0 destined for the AK ring. This is commonly used for credit returns and GO responses.

TxR_VERT_CYCLES_NE0

- **Title:**
- **Category:** Vertical Egress Events
- **Event Code:** 0x96
- **Register Restrictions :**
- **Definition:** Number of cycles the common mesh stop egress was not empty. The egress is used to queue up requests destined for the vertical ring on the mesh.

Table 2-313. Unit Masks for TxR_VERT_CYCLES_NE0

Extension	umask [15:8]	xtra [57:32]	Description
AD_AG0	bxxxxxxx1	bx1xxxxxxx xxxxxxxxxx xxxxxx	AD - Agent 0 Ring transactions from Agent 0 destined for the AD ring. Some example include outbound requests, snoop requests, and snoop responses.
AK_AG0	bxxxxxx1x	bx1xxxxxxx xxxxxxxxxx xxxxxx	AK - Agent 0 Ring transactions from Agent 0 destined for the AK ring. This is commonly used for credit returns and GO responses.
BL_AG0	bxxxxx1xx	bx1xxxxxxx xxxxxxxxxx xxxxxx	BL - Agent 0 Ring transactions from Agent 0 destined for the BL ring. This is commonly used to send data from the cache to various destinations.
IV_AG0	bxxxx1xxx	bx1xxxxxxx xxxxxxxxxx xxxxxx	IV - Agent 0 Ring transactions from Agent 0 destined for the IV ring. This is commonly used for snoops to the cores.
AD_AG1	bxxx1xxxx	bx1xxxxxxx xxxxxxxxxx xxxxxx	AD - Agent 1 Ring transactions from Agent 1 destined for the AD ring. This is commonly used for outbound requests.
AK_AG1	bxx1xxxxx	bx1xxxxxxx xxxxxxxxxx xxxxxx	AK - Agent 1 Ring transactions from Agent 1 destined for the AK ring.
BL_AG1	bx1xxxxxx	bx1xxxxxxx xxxxxxxxxx xxxxxx	BL - Agent 1 Ring transactions from Agent 1 destined for the BL ring. This is commonly used for transferring write back data to the cache.

TxR_VERT_CYCLES_NE1

- **Title:**
- **Category:** Vertical Egress Events
- **Event Code:** 0x97
- **Register Restrictions :**
- **Definition:** Number of cycles the common mesh stop egress was not empty. The egress is used to queue up requests destined for the vertical ring on the mesh.

Table 2-314. Unit Masks for TxR_VERT_CYCLES_NE1

Extension	umask [15:8]	xtra [57:32]	Description
AKC_AG0	bxxxxxxx1	bx1xxxxxxx xxxxxxxxxx xxxxxx	AKC - Agent 0 Ring transactions from Agent 0 destined for the AD ring. Some example include outbound requests, snoop requests, and snoop responses.
AKC_AG1	bxxxxxx1x	bx1xxxxxxx xxxxxxxxxx xxxxxx	AKC - Agent 1 Ring transactions from Agent 0 destined for the AK ring. This is commonly used for credit returns and GO responses.

TxR_VERT_INSERTS0

- **Title:**
- **Category:** Vertical Egress Events
- **Event Code:** 0x92
- **Register Restrictions :**
- **Definition:** Number of allocations into the common mesh stop egress. The egress is used to queue up requests destined for the vertical ring on the mesh.

Table 2-315. Unit Masks for TxR_VERT_INSERTS0

Extension	umask [15:8]	xtra [57:32]	Description
AD_AG0	bxxxxxxx1	bx1xxxxxxx xxxxxxxxxx xxxxxx	AD - Agent 0 Ring transactions from Agent 0 destined for the AD ring. Some example include outbound requests, snoop requests, and snoop responses.
AK_AG0	bxxxxxx1x	bx1xxxxxxx xxxxxxxxxx xxxxxx	AK - Agent 0 Ring transactions from Agent 0 destined for the AK ring. This is commonly used for credit returns and GO responses.
BL_AG0	bxxxxx1xx	bx1xxxxxxx xxxxxxxxxx xxxxxx	BL - Agent 0 Ring transactions from Agent 0 destined for the BL ring. This is commonly used to send data from the cache to various destinations.
IV_AG0	bxxxx1xxx	bx1xxxxxxx xxxxxxxxxx xxxxxx	IV - Agent 0 Ring transactions from Agent 0 destined for the IV ring. This is commonly used for snoops to the cores.
AD_AG1	bxxx1xxxx	bx1xxxxxxx xxxxxxxxxx xxxxxx	AD - Agent 1 Ring transactions from Agent 1 destined for the AD ring. This is commonly used for outbound requests.
AK_AG1	bxx1xxxxx	bx1xxxxxxx xxxxxxxxxx xxxxxx	AK - Agent 1 Ring transactions from Agent 1 destined for the AK ring.
BL_AG1	bx1xxxxxx	bx1xxxxxxx xxxxxxxxxx xxxxxx	BL - Agent 1 Ring transactions from Agent 1 destined for the BL ring. This is commonly used for transferring write back data to the cache.

TxR_VERT_INSERTS1

- **Title:**
- **Category:** Vertical Egress Events
- **Event Code:** 0x93
- **Register Restrictions :**
- **Definition:** Number of allocations into the common mesh stop egress. The egress is used to queue up requests destined for the vertical ring on the mesh.

Table 2-316. Unit Masks for TxR_VERT_INSERTS1

Extension	umask [15:8]	xtra [57:32]	Description
AKC_AG0	bxxxxxxx1	bx1xxxxxxx xxxxxxxxxx xxxxxx	AKC - Agent 0 Ring transactions from Agent 0 destined for the AD ring. Some example include outbound requests, snoop requests, and snoop responses.
AKC_AG1	bxxxxxx1x	bx1xxxxxxx xxxxxxxxxx xxxxxx	AKC - Agent 1 Ring transactions from Agent 0 destined for the AK ring. This is commonly used for credit returns and GO responses.

TxR_VERT_IRADS_USED

- **Title:**
- **Category:** Vertical Egress Events
- **Event Code:** 0x9F
- **Register Restrictions :**
- **Definition:**

Table 2-317. Unit Masks for TxR_VERT_IRADS_USED

Extension	umask [15:8]	xtra [57:32]	Description
AD_UNCRD	bxxxxxxx1	bx1xxxxxxx xxxxxxxxxx xxxxxx	
AD_CRD	bxxxxxx1x	bx1xxxxxxx xxxxxxxxxx xxxxxx	
BL_UNCRD	bxxxxx1xx	bx1xxxxxxx xxxxxxxxxx xxxxxx	
BL_CRD	bxxxx1xxx	bx1xxxxxxx xxxxxxxxxx xxxxxx	
AK	bxxx1xxxx	bx1xxxxxxx xxxxxxxxxx xxxxxx	
AKC	bxx1xxxxx	bx1xxxxxxx xxxxxxxxxx xxxxxx	
IV	bx1xxxxxx	bx1xxxxxxx xxxxxxxxxx xxxxxx	

TxR_VERT_NACK0

- **Title:**
- **Category:** Vertical Egress Events
- **Event Code:** 0x98
- **Register Restrictions :**
- **Definition:** Counts number of egress packets NACKed onto the vertical ring.

Table 2-318. Unit Masks for TxR_VERT_NACK0 (Sheet 1 of 2)

Extension	umask [15:8]	xtra [57:32]	Description
AD_AG0	bxxxxxxx1	bx1xxxxxxx xxxxxxxxxx xxxxxx	AD - Agent 0
AK_AG0	bxxxxxx1x	bx1xxxxxxx xxxxxxxxxx xxxxxx	AK - Agent 0
BL_AG0	bxxxxx1xx	bx1xxxxxxx xxxxxxxxxx xxxxxx	BL - Agent 0
IV_AG0	bxxxx1xxx	bx1xxxxxxx xxxxxxxxxx xxxxxx	IV
AD_AG1	bxxx1xxxx	bx1xxxxxxx xxxxxxxxxx xxxxxx	AD - Agent 1

Table 2-318. Unit Masks for TxR_VERT_NACK0 (Sheet 2 of 2)

Extension	umask [15:8]	xtra [57:32]	Description
AK_AG1	bxx1xxxxx	bx1xxxxxxx xxxxxxxxxx xxxxxx	AK - Agent 1
BL_AG1	bx1xxxxxx	bx1xxxxxxx xxxxxxxxxx xxxxxx	BL - Agent 1

TxR_VERT_NACK1

- **Title:**
- **Category:** Vertical Egress Events
- **Event Code:** 0x99
- **Register Restrictions :**
- **Definition:** Counts number of egress packets NACKed onto the vertical ring.

Table 2-319. Unit Masks for TxR_VERT_NACK1

Extension	umask [15:8]	xtra [57:32]	Description
AKC_AG0	bxxxxxxx1	bx1xxxxxxx xxxxxxxxxx xxxxxx	AKC - Agent 0
AKC_AG1	bxxxxxx1x	bx1xxxxxxx xxxxxxxxxx xxxxxx	AKC - Agent 1

TxR_VERT_OCCUPANCY0

- **Title:**
- **Category:** Vertical Egress Events
- **Event Code:** 0x90
- **Register Restrictions :**
- **Definition:** Occupancy event for the egress buffers in the common mesh stop. The egress is used to queue up requests destined for the vertical ring on the mesh.

Table 2-320. Unit Masks for TxR_VERT_OCCUPANCY0 (Sheet 1 of 2)

Extension	umask [15:8]	xtra [57:32]	Description
AD_AG0	bxxxxxxx1	bx1xxxxxxx xxxxxxxxxx xxxxxx	AD - Agent 0 Ring transactions from Agent 0 destined for the AD ring. Some example include outbound requests, snoop requests, and snoop responses.
AK_AG0	bxxxxxx1x	bx1xxxxxxx xxxxxxxxxx xxxxxx	AK - Agent 0 Ring transactions from Agent 0 destined for the AK ring. This is commonly used for credit returns and GO responses.
BL_AG0	bxxxxx1xx	bx1xxxxxxx xxxxxxxxxx xxxxxx	BL - Agent 0 Ring transactions from Agent 0 destined for the BL ring. This is commonly used to send data from the cache to various destinations.
IV_AG0	bxxxx1xxx	bx1xxxxxxx xxxxxxxxxx xxxxxx	IV - Agent 0 Ring transactions from Agent 0 destined for the IV ring. This is commonly used for snoops to the cores.

Table 2-320. Unit Masks for TxR_VERT_OCCUPANCY0 (Sheet 2 of 2)

Extension	umask [15:8]	xtra [57:32]	Description
AD_AG1	bxxx1xxxx	bx1xxxxxxx xxxxxxxxxx xxxxxx	AD - Agent 1 Ring transactions from Agent 1 destined for the AD ring. This is commonly used for outbound requests.
AK_AG1	bxx1xxxxx	bx1xxxxxxx xxxxxxxxxx xxxxxx	AK - Agent 1 Ring transactions from Agent 1 destined for the AK ring.
BL_AG1	bx1xxxxxx	bx1xxxxxxx xxxxxxxxxx xxxxxx	BL - Agent 1 Ring transactions from Agent 1 destined for the BL ring. This is commonly used for transferring write back data to the cache.

TxR_VERT_OCCUPANCY1

- **Title:**
- **Category:** Vertical Egress Events
- **Event Code:** 0x91
- **Register Restrictions :**
- **Definition:** Occupancy event for the egress buffers in the common mesh stop. The egress is used to queue up requests destined for the vertical ring on the mesh.

Table 2-321. Unit Masks for TxR_VERT_OCCUPANCY1

Extension	umask [15:8]	xtra [57:32]	Description
AKC_AG0	bxxxxxxx1	bx1xxxxxxx xxxxxxxxxx xxxxxx	AKC - Agent 0 Ring transactions from Agent 0 destined for the AD ring. Some example include outbound requests, snoop requests, and snoop responses.
AKC_AG1	bxxxxxx1x	bx1xxxxxxx xxxxxxxxxx xxxxxx	AKC - Agent 1 Ring transactions from Agent 0 destined for the AK ring. This is commonly used for credit returns and GO responses.

TxR_VERT_STARVED0

- **Title:**
- **Category:** Vertical Egress Events
- **Event Code:** 0x9A
- **Register Restrictions :**
- **Definition:** Counts injection starvation. This starvation is triggered when the CMS egress cannot send a transaction onto the vertical ring for a long period of time.

Table 2-322. Unit Masks for TxR_VERT_STARVED0 (Sheet 1 of 2)

Extension	umask [15:8]	xtra [57:32]	Description
AD_AG0	bxxxxxxx1	bx1xxxxxxx xxxxxxxxxx xxxxxx	AD - Agent 0
AK_AG0	bxxxxxx1x	bx1xxxxxxx xxxxxxxxxx xxxxxx	AK - Agent 0
BL_AG0	bxxxxx1xx	bx1xxxxxxx xxxxxxxxxx xxxxxx	BL - Agent 0

Table 2-322. Unit Masks for TxR_VERT_STARVED0 (Sheet 2 of 2)

Extension	umask [15:8]	xtra [57:32]	Description
IV_AG0	bxxxx1xxx	bx1xxxxxxx xxxxxxxxxx xxxxxx	IV
AD_AG1	bxxx1xxxx	bx1xxxxxxx xxxxxxxxxx xxxxxx	AD - Agent 1
AK_AG1	bxx1xxxxx	bx1xxxxxxx xxxxxxxxxx xxxxxx	AK - Agent 1
BL_AG1	bx1xxxxxx	bx1xxxxxxx xxxxxxxxxx xxxxxx	BL - Agent 1

TxR_VERT_STARVED1

- **Title:**
- **Category:** Vertical Egress Events
- **Event Code:** 0x9B
- **Register Restrictions :**
- **Definition:** Counts injection starvation. This starvation is triggered when the CMS egress cannot send a transaction onto the vertical ring for a long period of time.

Table 2-323. Unit Masks for TxR_VERT_STARVED1

Extension	umask [15:8]	xtra [57:32]	Description
AKC_AG0	bxxxxxxx1	bx1xxxxxxx xxxxxxxxxx xxxxxx	AKC - Agent 0
AKC_AG1	bxxxxxx1x	bx1xxxxxxx xxxxxxxxxx xxxxxx	AKC - Agent 1
TGC	bxxxxx1xx	bx1xxxxxxx xxxxxxxxxx xxxxxx	AKC - Agent 0

VERT_RING_AD_IN_USE

- **Title:**
- **Category:** Vertical In Use Ring Events
- **Event Code:** 0xB0
- **Register Restrictions :**
- **Definition:** Counts the number of cycles that the vertical AD ring is being used at this ring stop. This includes when packets are passing by and when packets are being sunk, but does not include when packets are being sent from the ring stop. We really have two rings -- a clockwise ring and a counter-clockwise ring. On the left side of the ring, the "UP" direction is on the clockwise ring and "DN" is on the counter-clockwise ring. On the right side of the ring, this is reversed. The first half of the CBos are on the left side of the ring, and the second half are on the right side of the ring. In other words (for example), in a 4c part, Cbo 0 UP AD is NOT the same ring as CBo 2 UP AD because they are on opposite sides of the ring.

Table 2-324. Unit Masks for VERT_RING_AD_IN_USE

Extension	umask [15:8]	xtra [57:32]	Description
UP_EVEN	bxxxxxxx1	bx1xxxxxxxxxxxxxxxxxxxxxxx	Up and Even
UP_ODD	bxxxxxx1x	bx1xxxxxxxxxxxxxxxxxxxxxxx	Up and Odd
DN_EVEN	bxxxxx1xx	bx1xxxxxxxxxxxxxxxxxxxxxxx	Down and Even
DN_ODD	bxxxx1xxx	bx1xxxxxxxxxxxxxxxxxxxxxxx	Down and Odd

VERT_RING_AKC_IN_USE

- **Title:**
- **Category:** Vertical In Use Ring Events
- **Event Code:** 0xB4
- **Register Restrictions :**
- **Definition:** Counts the number of cycles that the vertical AKC ring is being used at this ring stop. This includes when packets are passing by and when packets are being sunk, but does not include when packets are being sent from the ring stop. We really have two rings in JKT -- a clockwise ring and a counter-clockwise ring. On the left side of the ring, the “UP” direction is on the clockwise ring and “DN” is on the counter-clockwise ring. On the right side of the ring, this is reversed. The first half of the CBos are on the left side of the ring, and the second half are on the right side of the ring. In other words (for example), in a 4c part, Cbo 0 UP AD is NOT the same ring as CBo 2 UP AD because they are on opposite sides of the ring.

Table 2-325. Unit Masks for VERT_RING_AKC_IN_USE

Extension	umask [15:8]	xtra [57:32]	Description
UP_EVEN	bxxxxxxx1	bx1xxxxxxxxxxxxxxxxxxxxxxx	Up and Even
UP_ODD	bxxxxxx1x	bx1xxxxxxxxxxxxxxxxxxxxxxx	Up and Odd
DN_EVEN	bxxxxx1xx	bx1xxxxxxxxxxxxxxxxxxxxxxx	Down and Even
DN_ODD	bxxxx1xxx	bx1xxxxxxxxxxxxxxxxxxxxxxx	Down and Odd

VERT_RING_AK_IN_USE

- **Title:**
- **Category:** Vertical In Use Ring Events
- **Event Code:** 0xB1
- **Register Restrictions :**

- **Definition:** Counts the number of cycles that the vertical AK ring is being used at this ring stop. This includes when packets are passing by and when packets are being sunk, but does not include when packets are being sent from the ring stop. We really have two rings in -- a clockwise ring and a counter-clockwise ring. On the left side of the ring, the "UP" direction is on the clockwise ring and "DN" is on the counter-clockwise ring. On the right side of the ring, this is reversed. The first half of the CBos are on the left side of the ring, and the second half are on the right side of the ring. In other words (for example), in a 4c part, Cbo 0 UP AD is NOT the same ring as CBo 2 UP AD because they are on opposite sides of the ring.

Table 2-326. Unit Masks for VERT_RING_AK_IN_USE

Extension	umask [15:8]	xtra [57:32]	Description
UP_EVEN	bxxxxxxx1	bx1xxxxxxx xxxxxxxxxx xxxxxx	Up and Even
UP_ODD	bxxxxxx1x	bx1xxxxxxx xxxxxxxxxx xxxxxx	Up and Odd
DN_EVEN	bxxxxx1xx	bx1xxxxxxx xxxxxxxxxx xxxxxx	Down and Even
DN_ODD	bxxxx1xxx	bx1xxxxxxx xxxxxxxxxx xxxxxx	Down and Odd

VERT_RING_BL_IN_USE

- **Title:**
- **Category:** Vertical In Use Ring Events
- **Event Code:** 0xB2
- **Register Restrictions :**
- **Definition:** Counts the number of cycles that the vertical BL ring is being used at this ring stop. This includes when packets are passing by and when packets are being sunk, but does not include when packets are being sent from the ring stop. We really have two rings -- a clockwise ring and a counter-clockwise ring. On the left side of the ring, the "UP" direction is on the clockwise ring and "DN" is on the counter-clockwise ring. On the right side of the ring, this is reversed. The first half of the CBos are on the left side of the ring, and the second half are on the right side of the ring. In other words (for example), in a 4c part, Cbo 0 UP AD is NOT the same ring as CBo 2 UP AD because they are on opposite sides of the ring.

Table 2-327. Unit Masks for VERT_RING_BL_IN_USE

Extension	umask [15:8]	xtra [57:32]	Description
UP_EVEN	bxxxxxxx1	bx1xxxxxxx xxxxxxxxxx xxxxxx	Up and Even
UP_ODD	bxxxxxx1x	bx1xxxxxxx xxxxxxxxxx xxxxxx	Up and Odd
DN_EVEN	bxxxxx1xx	bx1xxxxxxx xxxxxxxxxx xxxxxx	Down and Even
DN_ODD	bxxxx1xxx	bx1xxxxxxx xxxxxxxxxx xxxxxx	Down and Odd

VERT_RING_IV_IN_USE

- **Title:**
- **Category:** Vertical In Use Ring Events
- **Event Code:** 0xB3
- **Register Restrictions :**
- **Definition:** Counts the number of cycles that the vertical IV ring is being used at this ring stop. This includes when packets are passing by and when packets are being sunk, but does not include when packets are being sent from the ring stop. There is only one IV ring. Therefore, if one wants to monitor the "Even" ring, they should select both UP_EVEN and DN_EVEN. To monitor the "Odd" ring, they should select both UP_ODD and DN_ODD.

Table 2-328. Unit Masks for VERT_RING_IV_IN_USE

Extension	umask [15:8]	xtra [57:32]	Description
UP	bxxxxxxx1	bx1xxxxxxx xxxxxxxxxx xxxxxx	Up
DN	bxxxxx1xx	bx1xxxxxxx xxxxxxxxxx xxxxxx	Down

VERT_RING_TGC_IN_USE

- **Title:**
- **Category:** Vertical In Use Ring Events
- **Event Code:** 0xB5
- **Register Restrictions :**
- **Definition:** Counts the number of cycles that the vertical TGC ring is being used at this ring stop. This includes when packets are passing by and when packets are being sunk, but does not include when packets are being sent from the ring stop. We really have two rings in JKT -- a clockwise ring and a counter-clockwise ring. On the left side of the ring, the "UP" direction is on the clockwise ring and "DN" is on the counter-clockwise ring. On the right side of the ring, this is reversed. The first half of the CBos are on the left side of the ring, and the second half are on the right side of the ring. In other words (for example), in a 4c part, Cbo 0 UP AD is NOT the same ring as CBo 2 UP AD because they are on opposite sides of the ring.

Table 2-329. Unit Masks for VERT_RING_TGC_IN_USE

Extension	umask [15:8]	xtra [57:32]	Description
UP_EVEN	bxxxxxxx1	bx1xxxxxxx xxxxxxxxxx xxxxxx	Up and Even
UP_ODD	bxxxxxx1x	bx1xxxxxxx xxxxxxxxxx xxxxxx	Up and Odd
DN_EVEN	bxxxxx1xx	bx1xxxxxxx xxxxxxxxxx xxxxxx	Down and Even
DN_ODD	bxxxx1xxx	bx1xxxxxxx xxxxxxxxxx xxxxxx	Down and Odd

TxC_AD_CREDITS

- **Title:**
- **Category:** AD CMS/Mesh Egress Credit Events
- **Event Code:** 0x08
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-330. Unit Masks for TxC_AD_CREDITS

Extension	umask [15:8]	Description
RETURNED	bxxxxxxx1	Credit Returns Counts the number of AD credits returns that m2m acquired from CMS.
ZERO	bxxxxxx1x	Zero Credits Counts the number of cycles AD egress credits is zero.
STALLED	bxxxxx1xx	Stalled, No Credits Counts the number of cycles AD egress is stalled due to non-availability of AD credits.

TxC_AD_CREDIT_OCCUPANCY

- **Title:**
- **Category:** AD CMS/Mesh Egress Credit Events
- **Event Code:** 0x9
- **Register Restrictions :** 0-3
- **Definition:** This event counts the number of AD egress Txns.

TxC_BL

- **Title:**
- **Category:** BL Egress Events
- **Event Code:** 0xE
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-331. Unit Masks for TxC_BL

Extension	umask [15:8]	xtra [57:32]	Description
INSERTS_CMS0	bxxxxxxx1	bxxxxxxx1	Inserts - CMS0 - Near Side Counts the number of BL transactions to CMS add port 0
INSERTS_CMS1	bxxxxxxx1	bxxxxxx1x	Inserts - CMS1 - Far Side Counts the number of BL transactions to CMS add port 1
NE_CMS0	bxxxxxx1x	bxxxxxxx1	Not Empty - CMS0 - Near Side Counts the cycles when BL egress is not empty where transactions are targeting add port 0
NE_CMS1	bxxxxxx1x	bxxxxxx1x	Not Empty - CMS1 - Far Side Counts the cycles when BL egress is not empty where transactions are targeting add port 1
FULL_CMS0	bxxxxx1xx	bxxxxxxx1	Full - CMS0 - Near Side Counts the cycles where BL egress is full for ADD 0 port

DISTRESS_ASSERTED

- **Title:** Distress signal asserted
- **Category:** Horizontal RING Events
- **Event Code:** 0xaf
- Max. Inc/Cyc:. 0, **Register Restrictions:**
- **Definition:** Counts the number of cycles either the local or incoming distress signals are asserted.

Table 2-332. Unit Masks for DISTRESS_ASSERTED

Extension	umask [15:8]	Description
VERT	b00000001	Vertical If IRQ egress is full, then agents will throttle outgoing AD IDI transactions
HORZ	b00000010	Horizontal If TGR egress is full, then agents will throttle outgoing AD IDI transactions
DPT_LOCAL	bxxxxx1xx	DPT Local Dynamic Prefetch Throttle triggered by this tile
DPT_NONLOCAL	bxxxx1xxx	DPT Remote Dynamic Prefetch Throttle received by this tile
PMM_LOCAL	bxxx1xxxx	DDRT Local If the CHA TOR has too many PMM transactions, this signal will throttle outgoing MS2IDI traffic
PMM_NONLOCAL	bxx1xxxxx	DDRT Remote If another CHA TOR has too many PMM transactions, this signal will throttle outgoing MS2IDI traffic
DPT_STALL_IV	bx1xxxxxx	DPT Stalled - IV DPT occurred while regular IVs were received, causing DPT to be stalled
DPT_STALL_NOCRD	b1xxxxxxx	DPT Stalled - No Credit DPT occurred while credit not available causing DPT to be stalled

2.8 M2PCIe* Performance Monitoring

M2PCIe* blocks manage the interface between the mesh and each IIO stack.

2.8.1 M2PCIe* Performance Monitoring Overview

Each M2PCIe* Box supports event monitoring through four 48b wide counters (M2n_PCI_PMON_CTR/CTL{3:0}). Each of these four counters can be programmed to count almost any M2PCIe* event.

Note: Only counter 0 can be used for tracking occupancy events.

2.8.2 M2PCIe* Performance Monitoring Events

M2PCIe* provides events to track information related to all the traffic passing through its boundaries.

- IIO credit tracking - credits rejected, acquired and used all broken down by message class.

2.8.3 M2PCIE Box Events Ordered By Code

The following table summarizes the directly measured M2PCIE Box events.

Table 2-333. Directly Measured M2PCIE Box Events

Symbol Name	Event Code	Ctrs	Description
NOTHING	0x0	0-3	
CLOCKTICKS	0x1	0-3	Clock ticks of the mesh to PCI (M2P)
RxC_CYCLES_NE	0x10	0-3	Ingress (from CMS) Queue Cycles Not Empty
RxC_INSERTS	0x11	0-3	Ingress (from CMS) Queue Inserts
P2P_CRD_OCCUPANCY	0x14	0-1	P2P Credit Occupancy
P2P_SHAR_RECEIVED	0x15	0-3	Shared Credits Received
P2P_DED_RECEIVED	0x16	0-3	Dedicated Credits Received
LOCAL_P2P_SHAR_RETURNED	0x17	0-3	Local P2P Shared Credits Returned
REMOTE_P2P_SHAR_RETURNED	0x18	0-3	Remote P2P Shared Credits Returned
LOCAL_P2P_DED_RETURNED_0	0x19	0-3	Local P2P Dedicated Credits Returned - 0
LOCAL_P2P_DED_RETURNED_1	0x1A	0-3	Local P2P Dedicated Credits Returned - 1
REMOTE_P2P_DED_RETURNED	0x1B	0-3	Remote P2P Dedicated Credits Returned
TxC_CYCLES_NE	0x23	0-1	Egress (to CMS) Cycles Not Empty
TxC_CYCLES_FULL	0x25	0-3	Egress (to CMS) Cycles Full
TxC_CREDITS	0x2D	0-1	Egress Credits
IIO_CREDITS_USED	0x32	0-3	M2PCIe IIO Credits in Use
IIO_CREDITS_ACQUIRED	0x33	0-3	M2PCIe IIO Credit Acquired
IIO_CREDITS_REJECT	0x34	0-3	M2PCIe IIO Failed to Acquire a Credit
LOCAL_SHAR_P2P_CRD_TAKEN_0	0x40	0-3	Local Shared P2P Credit Taken - 0
LOCAL_SHAR_P2P_CRD_TAKEN_1	0x41	0-3	Local Shared P2P Credit Taken - 1
REMOTE_SHAR_P2P_CRD_TAKEN_0	0x42	0-3	Remote Shared P2P Credit Taken - 0
REMOTE_SHAR_P2P_CRD_TAKEN_1	0x43	0-3	Remote Shared P2P Credit Taken - 1
LOCAL_SHAR_P2P_CRD_RETURNED	0x44	0-3	Local Shared P2P Credit Returned to credit ring
REMOTE_SHAR_P2P_CRD_RETURNED	0x45	0-3	Remote Shared P2P Credit Returned to credit ring
LOCAL_DED_P2P_CRD_TAKEN_0	0x46	0-3	Local Dedicated P2P Credit Taken - 0
LOCAL_DED_P2P_CRD_TAKEN_1	0x47	0-3	Local Dedicated P2P Credit Taken - 1
REMOTE_DED_P2P_CRD_TAKEN_0	0x48	0-3	Remote Dedicated P2P Credit Taken - 0
REMOTE_DED_P2P_CRD_TAKEN_1	0x49	0-3	Remote Dedicated P2P Credit Taken - 1
LOCAL_SHAR_P2P_CRD_WAIT_0	0x4A	0-3	Waiting on Local Shared P2P Credit - 0
LOCAL_SHAR_P2P_CRD_WAIT_1	0x4B	0-3	Waiting on Local Shared P2P Credit - 1
REMOTE_SHAR_P2P_CRD_WAIT_0	0x4C	0-3	Waiting on Remote Shared P2P Credit - 0
REMOTE_SHAR_P2P_CRD_WAIT_1	0x4D	0-3	Waiting on Remote Shared P2P Credit - 1
TxC_INSERTS_AD	0x50		Egress (to CMS) Ingress - AD
TxC_INSERTS_AK	0x51		Egress (to CMS) Ingress - AK

2.8.4 M2PCIE Box Performance Monitor Event List

The section enumerates the 5th Gen Intel® Xeon® Scalable Processor performance monitoring events for the M2PCIE box.

CLOCKTICKS

- **Title:**
- **Category:** Transgress Credit Events
- **Event Code:** 0x1
- **Register Restrictions :** 0-3
- **Definition:** Counts the number of UCLKs in the M3 UCLK domain. This could be slightly different than the count in the Ubox because of enable or freeze delays. However, because the M3 is close to the Ubox, they generally should not diverge by more than a handful of cycles.

IIO_CREDITS_ACQUIRED

- **Title:**
- **Category:** IIO Credit Events
- **Event Code:** 0x33
- **Register Restrictions :** 0-3
- **Definition:** Counts the number of credits that are acquired in the M2PCIe agent for sending transactions into the IIO on either NCB or NCS are in use. Transactions from the BL ring going into the IIO agent must first acquire a credit. These credits are for either the NCB or NCS message classes. NCB, or non-coherent bypass messages are used to transmit data without coherency (and are common). NCS is used for reads to PCIe (and should be used sparingly).

Table 2-334. Unit Masks for IIO_CREDITS_ACQUIRED

Extension	umask [15:8]	Description
DRS_0	bxxxxxxx1	DRS Credits for transfer through CMS Port 0 to the IIO for the DRS message class.
DRS_1	bxxxxxx1x	DRS Credits for transfer through CMS Port 0 to the IIO for the DRS message class.
NCB_0	bxxxxx1xx	NCB Credits for transfer through CMS Port 0 to the IIO for the NCB message class.
NCB_1	bxxxx1xxx	NCB Credits for transfer through CMS Port 0 to the IIO for the NCB message class.
NCS_0	bxxx1xxxx	NCS Credits for transfer through CMS Port 0 to the IIO for the NCS message class.
NCS_1	bxx1xxxxx	NCS Credit for transfer through CMS Port 0s to the IIO for the NCS message class.

IIO_CREDITS_REJECT

- **Title:**
- **Category:** IIO Credit Events
- **Event Code:** 0x34
- **Register Restrictions :** 0-3
- **Definition:** Counts the number of times that a request pending in the BL ingress attempted to acquire either a NCB or NCS credit to transmit into the IIO, but was rejected because no credits were available. NCB, or non-coherent bypass messages are used to transmit data without coherency (and are common). NCS is used for reads to PCIe (and should be used sparingly).

Table 2-335. Unit Masks for IIO_CREDITS_REJECT

Extension	umask [15:8]	Description
DRS	bxxxx1xxx	DRS Credits to the IIO for the DRS message class.
NCB	bxxx1xxxx	NCB Credits to the IIO for the NCB message class.
NCS	bxx1xxxxx	NCS Credits to the IIO for the NCS message class.

IIO_CREDITS_USED

- **Title:**
- **Category:** IIO Credit Events
- **Event Code:** 0x32
- **Register Restrictions :** 0-3
- **Definition:** Counts the number of cycles when one or more credits in the M2PCIe agent for sending transactions into the IIO on either NCB or NCS are in use. Transactions from the BL ring going into the IIO agent must first acquire a credit. These credits are for either the NCB or NCS message classes. NCB, or non-coherent bypass messages are used to transmit data without coherency (and are common). NCS is used for reads to PCIe (and should be used sparingly).

Table 2-336. Unit Masks for IIO_CREDITS_USED (Sheet 1 of 2)

Extension	umask [15:8]	Description
DRS_0	bxxxxxxx1	DRS to CMS Port 0 Credits for transfer through CMS Port 0 to the IIO for the DRS message class.
DRS_1	bxxxxxx1x	DRS to CMS Port 1 Credits for transfer through CMS Port 0 to the IIO for the DRS message class.
NCB_0	bxxxxx1xx	NCB to CMS Port 0 Credits for transfer through CMS Port 0 to the IIO for the NCB message class.
NCB_1	bxxxx1xxx	NCB to CMS Port 1 Credits for transfer through CMS Port 0 to the IIO for the NCB message class.

Table 2-336. Unit Masks for IIO_CREDITS_USED (Sheet 2 of 2)

Extension	umask [15:8]	Description
NCS_0	bxxx1xxxx	NCS to CMS Port 0 Credits for transfer through CMS Port 0 to the IIO for the NCS message class.
NCS_1	bxx1xxxxx	NCS to CMS Port 1 Credit for transfer through CMS Port 0s to the IIO for the NCS message class.

LOCAL_DED_P2P_CRD_TAKEN_0

- **Title:**
- **Category:** Egress P2P Credit Events
- **Event Code:** 0x46
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-337. Unit Masks for LOCAL_DED_P2P_CRD_TAKEN_0

Extension	umask [15:8]	Description
M2IOSF0_NCB	bxxxxxxx1	M2IOSF0 - NCB
M2IOSF0_NCS	bxxxxxx1x	M2IOSF0 - NCS
M2IOSF1_NCB	bxxxxx1xx	M2IOSF1 - NCB
M2IOSF1_NCS	bxxxx1xxx	M2IOSF1 - NCS
M2IOSF2_NCB	bxxx1xxxx	M2IOSF2 - NCB
M2IOSF2_NCS	bxx1xxxxx	M2IOSF2 - NCS
M2IOSF3_NCB	bx1xxxxxx	M2IOSF3 - NCB
M2IOSF3_NCS	b1xxxxxxx	M2IOSF3 - NCS

LOCAL_DED_P2P_CRD_TAKEN_1

- **Title:**
- **Category:** Egress P2P Credit Events
- **Event Code:** 0x47
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-338. Unit Masks for LOCAL_DED_P2P_CRD_TAKEN_1

Extension	umask [15:8]	Description
M2IOSF4_NCB	bxxxxxxx1	M2IOSF4 - NCB
M2IOSF4_NCS	bxxxxxx1x	M2IOSF4 - NCS
M2IOSF5_NCB	bxxxxx1xx	M2IOSF5 - NCB
M2IOSF5_NCS	bxxxx1xxx	M2IOSF5 - NCS

LOCAL_P2P_DED_RETURNED_0

- **Title:**
- **Category:** Ingress P2P Credit Events
- **Event Code:** 0x19
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-339. Unit Masks for LOCAL_P2P_DED_RETURNED_0

Extension	umask [15:8]	Description
MS2IOSF0_NCB	bxxxxxxx1	M2IOSF0 - NCB
MS2IOSF0_NCS	bxxxxxx1x	M2IOSF0 - NCS
MS2IOSF1_NCB	bxxxxx1xx	M2IOSF1 - NCB
MS2IOSF1_NCS	bxxxx1xxx	M2IOSF1 - NCS
MS2IOSF2_NCB	bxxx1xxxx	M2IOSF2 - NCB
MS2IOSF2_NCS	bxx1xxxxx	M2IOSF2 - NCS
MS2IOSF3_NCB	bx1xxxxxx	M2IOSF3 - NCB
MS2IOSF3_NCS	b1xxxxxxx	M2IOSF3 - NCS

LOCAL_P2P_DED_RETURNED_1

- **Title:**
- **Category:** Ingress P2P Credit Events
- **Event Code:** 0x1A
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-340. Unit Masks for LOCAL_P2P_DED_RETURNED_1

Extension	umask [15:8]	Description
MS2IOSF4_NCB	bxxxxxxx1	M2IOSF4 - NCB
MS2IOSF4_NCS	bxxxxxx1x	M2IOSF4 - NCS
MS2IOSF5_NCB	bxxxxx1xx	M2IOSF5 - NCB
MS2IOSF5_NCS	bxxxx1xxx	M2IOSF5 - NCS

LOCAL_P2P_SHAR_RETURNED

- **Title:**
- **Category:** Ingress P2P Credit Events
- **Event Code:** 0x17
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-341. Unit Masks for LOCAL_P2P_SHAR_RETURNED

Extension	umask [15:8]	Description
AGENT_0	bxxxxxxx1	Agent0
AGENT_1	bxxxxxx1x	Agent1
AGENT_2	bxxxxx1xx	Agent2

LOCAL_SHAR_P2P_CRD_RETURNED

- **Title:**
- **Category:** Egress P2P Credit Events
- **Event Code:** 0x44
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-342. Unit Masks for LOCAL_SHAR_P2P_CRD_RETURNED (Sheet 1 of 2)

Extension	umask [15:8]	Description
AGENT_0	bxxxxxxx1	Agent0
AGENT_1	bxxxxxx1x	Agent1
AGENT_2	bxxxxx1xx	Agent2
AGENT_3	bxxxx1xxx	Agent3

Table 2-342. Unit Masks for LOCAL_SHAR_P2P_CRD_RETURNED (Sheet 2 of 2)

Extension	umask [15:8]	Description
AGENT_4	bxxx1xxxx	Agent4
AGENT_5	bxx1xxxxx	Agent5

LOCAL_SHAR_P2P_CRD_TAKEN_0

- **Title:**
- **Category:** Egress P2P Credit Events
- **Event Code:** 0x40
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-343. Unit Masks for LOCAL_SHAR_P2P_CRD_TAKEN_0

Extension	umask [15:8]	Description
M2IOSF0_NCB	bxxxxxxx1	M2IOSF0 - NCB
M2IOSF0_NCS	bxxxxxx1x	M2IOSF0 - NCS
M2IOSF1_NCB	bxxxxx1xx	M2IOSF1 - NCB
M2IOSF1_NCS	bxxxx1xxx	M2IOSF1 - NCS
M2IOSF2_NCB	bxxx1xxxx	M2IOSF2 - NCB
M2IOSF2_NCS	bxx1xxxxx	M2IOSF2 - NCS
M2IOSF3_NCB	bx1xxxxxx	M2IOSF3 - NCB
M2IOSF3_NCS	b1xxxxxxx	M2IOSF3 - NCS

LOCAL_SHAR_P2P_CRD_TAKEN_1

- **Title:**
- **Category:** Egress P2P Credit Events
- **Event Code:** 0x41
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-344. Unit Masks for LOCAL_SHAR_P2P_CRD_TAKEN_1 (Sheet 1 of 2)

Extension	umask [15:8]	Description
M2IOSF4_NCB	bxxxxxxx1	M2IOSF4 - NCB
M2IOSF4_NCS	bxxxxxx1x	M2IOSF4 - NCS

Table 2-344. Unit Masks for LOCAL_SHAR_P2P_CRD_TAKEN_1 (Sheet 2 of 2)

Extension	umask [15:8]	Description
M2IOSF5_NCB	bxxxxx1xx	M2IOSF5 - NCB
M2IOSF5_NCS	bxxxx1xxx	M2IOSF5 - NCS

LOCAL_SHAR_P2P_CRD_WAIT_0

- **Title:**
- **Category:** Egress P2P Credit Events
- **Event Code:** 0x4A
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-345. Unit Masks for LOCAL_SHAR_P2P_CRD_WAIT_0

Extension	umask [15:8]	Description
M2IOSF0_NCB	bxxxxxxx1	M2IOSF0 - NCB
M2IOSF0_NCS	bxxxxxx1x	M2IOSF0 - NCS
M2IOSF1_NCB	bxxxxx1xx	M2IOSF1 - NCB
M2IOSF1_NCS	bxxxx1xxx	M2IOSF1 - NCS
M2IOSF2_NCB	bxxx1xxxx	M2IOSF2 - NCB
M2IOSF2_NCS	bxx1xxxxx	M2IOSF2 - NCS
M2IOSF3_NCB	bx1xxxxxx	M2IOSF3 - NCB
M2IOSF3_NCS	b1xxxxxxx	M2IOSF3 - NCS

LOCAL_SHAR_P2P_CRD_WAIT_1

- **Title:**
- **Category:** Egress P2P Credit Events
- **Event Code:** 0x4B
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-346. Unit Masks for LOCAL_SHAR_P2P_CRD_WAIT_1 (Sheet 1 of 2)

Extension	umask [15:8]	Description
M2IOSF4_NCB	bxxxxxxx1	M2IOSF4 - NCB
M2IOSF4_NCS	bxxxxxx1x	M2IOSF4 - NCS

Table 2-346. Unit Masks for LOCAL_SHAR_P2P_CRD_WAIT_1 (Sheet 2 of 2)

Extension	umask [15:8]	Description
M2IOSF5_NCB	bxxxxx1xx	M2IOSF5 - NCB
M2IOSF5_NCS	bxxxx1xxx	M2IOSF5 - NCS

NOTHING

- **Title:**
- **Category:** Transgress Credit Events
- **Event Code:** 0x0
- **Register Restrictions :** 0-3
- **Definition:**

P2P_CRD_OCCUPANCY

- **Title:**
- **Category:** Ingress P2P Credit Events
- **Event Code:** 0x14
- **Register Restrictions :** 0-1
- **Definition:**

Table 2-347. Unit Masks for P2P_CRD_OCCUPANCY

Extension	umask [15:8]	Description
LOCAL_NCB	bxxxxxxx1	Local NCB
LOCAL_NCS	bxxxxxx1x	Local NCS
REMOTE_NCB	bxxxxx1xx	Remote NCB
REMOTE_NCS	bxxxx1xxx	Remote NCS
ALL	bxxx1xxxx	All

P2P_DED_RECEIVED

- **Title:**
- **Category:** Ingress P2P Credit Events
- **Event Code:** 0x16
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-348. Unit Masks for P2P_DED_RECEIVED

Extension	umask [15:8]	Description
LOCAL_NCB	bxxxxxxx1	Local NCB
LOCAL_NCS	bxxxxxx1x	Local NCS
REMOTE_NCB	bxxxxx1xx	Remote NCB
REMOTE_NCS	bxxxx1xxx	Remote NCS
ALL	bxxx1xxxx	All

P2P_SHAR_RECEIVED

- **Title:**
- **Category:** Ingress P2P Credit Events
- **Event Code:** 0x15
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-349. Unit Masks for P2P_SHAR_RECEIVED

Extension	umask [15:8]	Description
LOCAL_NCB	bxxxxxxx1	Local NCB
LOCAL_NCS	bxxxxxx1x	Local NCS
REMOTE_NCB	bxxxxx1xx	Remote NCB
REMOTE_NCS	bxxxx1xxx	Remote NCS
ALL	bxxx1xxxx	All

REMOTE_DED_P2P_CRD_TAKEN_0

- **Title:**
- **Category:** Egress P2P Credit Events
- **Event Code:** 0x48
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-350. Unit Masks for REMOTE_DED_P2P_CRD_TAKEN_0

Extension	umask [15:8]	Description
UPI0_DRS	bxxxxxxx1	UPI0 - DRS
UPI0_NCB	bxxxxxx1x	UPI0 - NCB
UPI0_NCS	bxxxxx1xx	UPI0 - NCS
UPI1_DRS	bxxxx1xxx	UPI1 - DRS
UPI1_NCB	bxxx1xxxx	UPI1 - NCB
UPI1_NCS	bxx1xxxxx	UPI1 - NCS

REMOTE_DED_P2P_CRD_TAKEN_1

- **Title:**
- **Category:** Egress P2P Credit Events
- **Event Code:** 0x49
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-351. Unit Masks for REMOTE_DED_P2P_CRD_TAKEN_1

Extension	umask [15:8]	Description
UPI2_DRS	bxxxxxxx1	UPI2 - DRS
UPI2_NCB	bxxxxxx1x	UPI2 - NCB
UPI2_NCS	bxxxxx1xx	UPI2 - NCS

REMOTE_P2P_DED_RETURNED

- **Title:**
- **Category:** Ingress P2P Credit Events
- **Event Code:** 0x1B
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-352. Unit Masks for REMOTE_P2P_DED_RETURNED (Sheet 1 of 2)

Extension	umask [15:8]	Description
UPI0_NCB	bxxxxxxx1	UPI0 - NCB
UPI0_NCS	bxxxxxx1x	UPI0 - NCS

Table 2-352. Unit Masks for REMOTE_P2P_DED_RETURNED (Sheet 2 of 2)

Extension	umask [15:8]	Description
UPI1_NCB	bxxxxx1xx	UPI1 - NCB
UPI1_NCS	bxxxx1xxx	UPI1 - NCS
UPI2_NCB	bxxx1xxxx	UPI2 - NCB
UPI2_NCS	bxx1xxxxx	UPI2 - NCS

REMOTE_P2P_SHAR_RETURNED

- **Title:**
- **Category:** Ingress P2P Credit Events
- **Event Code:** 0x18
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-353. Unit Masks for REMOTE_P2P_SHAR_RETURNED

Extension	umask [15:8]	Description
AGENT_0	bxxxxxxx1	Agent0
AGENT_1	bxxxxxx1x	Agent1
AGENT_2	bxxxxx1xx	Agent2

REMOTE_SHAR_P2P_CRD_RETURNED

- **Title:**
- **Category:** Egress P2P Credit Events
- **Event Code:** 0x45
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-354. Unit Masks for REMOTE_SHAR_P2P_CRD_RETURNED

Extension	umask [15:8]	Description
AGENT_0	bxxxxxxx1	Agent0
AGENT_1	bxxxxxx1x	Agent1
AGENT_2	bxxxxx1xx	Agent2

REMOTE_SHAR_P2P_CRD_TAKEN_0

- **Title:**
- **Category:** Egress P2P Credit Events
- **Event Code:** 0x42
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-355. Unit Masks for REMOTE_SHAR_P2P_CRD_TAKEN_0

Extension	umask [15:8]	Description
UPI0_DRS	bxxxxxxx1	UPI0 - DRS
UPI0_NCB	bxxxxxx1x	UPI0 - NCB
UPI0_NCS	bxxxxx1xx	UPI0 - NCS
UPI1_DRS	bxxxx1xxx	UPI1 - DRS
UPI1_NCB	bxxx1xxxx	UPI1 - NCB
UPI1_NCS	bxx1xxxxx	UPI1 - NCS

REMOTE_SHAR_P2P_CRD_TAKEN_1

- **Title:**
- **Category:** Egress P2P Credit Events
- **Event Code:** 0x43
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-356. Unit Masks for REMOTE_SHAR_P2P_CRD_TAKEN_1

Extension	umask [15:8]	Description
UPI2_DRS	bxxxxxxx1	UPI2 - DRS
UPI2_NCB	bxxxxxx1x	UPI2 - NCB
UPI2_NCS	bxxxxx1xx	UPI2 - NCS

REMOTE_SHAR_P2P_CRD_WAIT_0

- **Title:**
- **Category:** Egress P2P Credit Events
- **Event Code:** 0x4C
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-357. Unit Masks for REMOTE_SHAR_P2P_CRD_WAIT_0

Extension	umask [15:8]	Description
UPI0_DRS	bxxxxxxx1	UPI0 - DRS
UPI0_NCB	bxxxxxx1x	UPI0 - NCB
UPI0_NCS	bxxxxx1xx	UPI0 - NCS
UPI1_DRS	bxxxx1xxx	UPI1 - DRS
UPI1_NCB	bxxx1xxxx	UPI1 - NCB
UPI1_NCS	bxx1xxxxx	UPI1 - NCS

REMOTE_SHAR_P2P_CRD_WAIT_1

- **Title:**
- **Category:** Egress P2P Credit Events
- **Event Code:** 0x4D
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-358. Unit Masks for REMOTE_SHAR_P2P_CRD_WAIT_1

Extension	umask [15:8]	Description
UPI2_DRS	bxxxxxxx1	UPI2 - DRS
UPI2_NCB	bxxxxxx1x	UPI2 - NCB
UPI2_NCS	bxxxxx1xx	UPI2 - NCS

RxC_CYCLES_NE

- **Title:**
- **Category:** Ingress Events
- **Event Code:** 0x10
- **Register Restrictions :** 0-3
- **Definition:** Counts the number of cycles when the M2PCIe ingress is not empty.

Table 2-359. Unit Masks for RxC_CYCLES_NE (Sheet 1 of 2)

Extension	umask [15:8]	Description
CHA_IDI	bxxxxxxx1	
CHA_NCB	bxxxxxx1x	

Table 2-359. Unit Masks for RxC_CYCLES_NE (Sheet 2 of 2)

Extension	umask [15:8]	Description
CHA_NCS	bxxxxx1xx	
UPI_NCB	bxxxx1xxx	
UPI_NCS	bxxx1xxxx	
IIO_NCB	bxx1xxxxx	
IIO_NCS	bx1xxxxxx	
ALL	b1xxxxxxx	

RxC_INSERTS

- **Title:**
- **Category:** Ingress Events
- **Event Code:** 0x11
- **Register Restrictions :** 0-3
- **Definition:** Counts the number of entries inserted into the M2PCIe ingress queue. This can be used in conjunction with the *M2PCIe Ingress Occupancy Accumulator* event in order to calculate average queue latency.

Table 2-360. Unit Masks for RxC_INSERTS

Extension	umask [15:8]	Description
CHA_IDI	bxxxxxxx1	
CHA_NCB	bxxxxxx1x	
CHA_NCS	bxxxxx1xx	
UPI_NCB	bxxxx1xxx	
UPI_NCS	bxxx1xxxx	
IIO_NCB	bxx1xxxxx	
IIO_NCS	bx1xxxxxx	
ALL	b1xxxxxxx	

TxC_CREDITS

- **Title:**
- **Category:** Egress Events
- **Event Code:** 0x2D
- **Register Restrictions :** 0-1
- **Definition:**

Table 2-361. Unit Masks for TxC_CREDITS

Extension	umask [15:8]	Description
PRQ	bxxxxxxx1	
PMM	bxxxxxx1x	

TxC_CYCLES_FULL

- **Title:**
- **Category:** Egress Events
- **Event Code:** 0x25
- **Register Restrictions :** 0-3
- **Definition:** Counts the number of cycles when the M2PCIe egress is full. This tracks messages for one of the two CMS ports that are used by the M2PCIe agent.

Table 2-362. Unit Masks for TxC_CYCLES_FULL

Extension	umask [15:8]	Description
PMM_BLOCK_1	bxxxx1xxx	
PMM_BLOCK_0	b1xxxxxxx	

TxC_CYCLES_NE

- **Title:**
- **Category:** Egress Events
- **Event Code:** 0x23
- **Register Restrictions :** 0-1
- **Definition:** Counts the number of cycles when the M2PCIe egress is not empty. This tracks messages for one of the two CMS ports that are used by the M2PCIe agent. This can be used in conjunction with the *M2PCIe Ingress Occupancy Accumulator* event in order to calculate average queue occupancy. Multiple egress buffers can be tracked at a given time using multiple counters.

Table 2-363. Unit Masks for TxC_CYCLES_NE

Extension	umask [15:8]	Description
PMM_DISTRESS_1	bxxxx1xxx	
PMM_DISTRESS_0	b1xxxxxxx	

TxC_INSERTS_AD

- **Title:**
- **Category:** Egress Events
- **Event Code:** 0x50
- **Register Restrictions :**

- **Definition:**

Table 2-364. Unit Masks for TxC_INSERTS_AD

Extension	umask [15:8]	Description
AGT_0	bxxxxxxx1	AD_0 CXL
AGT_1	bxxxxxx1x	AD_0 IO Coh Req
AGT_2	bxxxxx1xx	AD_0 IO Remote NDR
AGT_3	bxxxx1xxx	AD_1 CXL.
AGT_4	bxxx1xxxx	AD_1 IO Coh Req
AGT_5	bxx1xxxxx	AD_1 IO Remote NDR

TxC_INSERTS_AK

- **Title:**
- **Category:** Egress Events
- **Event Code:** 0x51
- **Register Restrictions :**
- **Definition:**

Table 2-365. Unit Masks for TxC_INSERTS_AK

Extension	umask [15:8]	Description
AGT_0	bxxxxxxx1	AK_0 VN0 Crd Rtn
AGT_1	bxxxxxx1x	AK_0 IO Rsp
AGT_2	bxxxxx1xx	AK_0 IO Go-I
AGT_3	bxxxx1xxx	AK_0 CXL.
AGT_4	bxxx1xxxx	AK_1 VN0 Crd Rtn
AGT_5	bxx1xxxxx	AK_1 CXL.Mem NDR

TxC_INSERTS_BL

- **Title:**
- **Category:** Egress Events
- **Event Code:** 0x52
- **Register Restrictions :**
- **Definition:**

Table 2-366. Unit Masks for TxC_INSERTS_BL

Extension	umask [15:8]	Description
AGT_0	bxxxxxxx1	BL IO DRS
AGT_1	bxxxxxx1x	BL IO NCB
AGT_2	bxxxxx1xx	BL IO NCS
AGT_3	bxxxx1xxx	BL CXL.
AGT_4	bxxx1xxxx	BL CXL.Mem Data

2.9 M3UPI Performance Monitoring

M3UPI is the interface between the mesh and the Intel® UPI link layer. It is responsible for translating between mesh protocol packets and flits that are used for transmitting data across the Intel® UPI interface. It performs credit checking between the local Intel® UPI LL, the remote Intel® UPI LL and other agents on the local mesh.

The M3UPI agent provides several functions:

- Interface between mesh and Intel® UPI:

  One of the primary attributes of the mesh is its ability to convey Intel® UPI semantics with no translation. For example, this architecture enables initiators to communicate with a local Home Agent (HA) in exactly the same way as a remote HA on another processor socket. With this philosophy, the M3UPI block is lean and does very little with regards to the Intel® UPI protocol aside from mirror the request between the mesh and the Intel® UPI interface.

- Intel® UPI routing:

  In order to optimize layout and latency, both full width Intel® UPI interfaces share the same mesh stop. Therefore, a Intel® UPI packet might be received on one interface and simply forwarded along on the other Intel® UPI interface. The M3UPI has sufficient routing logic to determine if a request, snoop or response is targeting the local socket or if it should be forwarded along to the other interface. This routing remains isolated to M3UPI and does not impede traffic on the mesh.

- Intel® UPI Home Snoop Protocol (with early snoop optimizations for DP):

  The M3UPI agent implements a latency-reducing optimization for dual sockets which issues snoops within the socket for incoming requests as well as a latency-reducing optimization to return data satisfying Direct2Core (D2C) requests.

2.9.1 M3UPI Performance Monitoring Overview

Each M3UPI link in supports event monitoring through four 48b wide counters (M3_Ly_PCI_PMON_CTR/CTL{3:0}). Each of these four counters can be programmed to count almost any M3UPI event.

Note: We have a restriction where it is not possible to count CMS and non-CMS events at the same time within a M3UPI instance.

2.9.2 M3UPI Box Events Ordered By Code

The following table summarizes the directly measured M3UPI box events.

Table 2-367. Directly Measured M3UPI Box Events (Sheet 1 of 2)

Symbol Name	Event Code	Ctrs	Description
CLOCKTICKS	0x1	0-3	Number of UCLKs in domain
TxC_AD_FLQ_OCCUPANCY	0x1C	0	AD Flow Q Occupancy
TxC_BL_FLQ_OCCUPANCY	0x1D	0	BL Flow Q Occupancy
TxC_AK_FLQ_OCCUPANCY	0x1E	0	AK Flow Q Occupancy
TxC_BL_WB_FLQ_OCCUPANCY	0x1F	0	BL Flow Q Occupancy
UPI_PEER_AD_CREDITS_EMPTY	0x20	0-3	UPI0 AD Credits Empty
UPI_PEER_BL_CREDITS_EMPTY	0x21	0-3	UPI0 BL Credits Empty
CHA_AD_CREDITS_EMPTY	0x22	0-3	CBox AD Credits Empty
M2_BL_CREDITS_EMPTY	0x23	0-3	M2 BL Credits Empty
TxC_AD_FLQ_CYCLES_NE	0x27	0-3	AD Flow Q Not Empty
TxC_BL_FLQ_CYCLES_NE	0x28	0-3	BL Flow Q Not Empty
UPI_PREFETCH_SPAWN	0x29	0-3	Flow Q Generated Prefetch
D2U_SENT	0x2A	0-3	D2U Sent
D2C_SENT	0x2B	0-3	D2C Sent
TxC_AD_FLQ_BYPASS	0x2C	0-3	AD Flow Q Bypass
TxC_AD_FLQ_INSERTS	0x2D	0-3	AD Flow Q Inserts
TxC_BL_FLQ_INSERTS	0x2E	0-3	BL Flow Q Inserts
TxC_AK_FLQ_INSERTS	0x2F	0-3	AK Flow Q Inserts
TxC_AD_ARB_FAIL	0x30	0-3	Failed ARB for AD
TxC_BL_ARB_FAIL	0x35	0-3	Failed ARB for BL
MULTI_SLOT_RCVD	0x3E	0-3	Multi Slot Flit Received
RxC_BYPASSED	0x40	0-2	Ingress Queue Bypasses
RxC_INSERTS_VN0	0x41	0-3	VN0 Ingress (from CMS) Queue - Inserts
RxC_INSERTS_VN1	0x42	0-3	VN1 Ingress (from CMS) Queue - Inserts
RxC_CYCLES_NE_VN0	0x43	0-3	VN0 Ingress (from CMS) Queue - Cycles Not Empty
RxC_CYCLES_NE_VN1	0x44	0-3	VN1 Ingress (from CMS) Queue - Cycles Not Empty
RxC_OCCUPANCY_VN0	0x45	0-3	VN0 Ingress (from CMS) Queue - Occupancy
RxC_OCCUPANCY_VN1	0x46	0-3	VN1 Ingress (from CMS) Queue - Occupancy
RxC_ARB_NOCRD_VN0	0x47	0-3	No Credits to Arb for VN0
RxC_ARB_NOCRD_VN1	0x48	0-3	No Credits to Arb for VN1
RxC_ARB_NOREQ_VN0	0x49	0-3	CantArbforVN0

Table 2-367. Directly Measured M3UPI Box Events (Sheet 2 of 2)

Symbol Name	Event Code	Ctrs	Description
RxC_ARB_NOREQ_VN1	0x4A	0-3	CantArbforVN1'
RxC_ARB_LOST_VN0	0x4B	0-3	Lost Arb for VN0
RxC_ARB_LOST_VN1	0x4C	0-3	Lost Arb for VN1
RxC_ARB_MISC	0x4D	0-3	Arb Miscellaneous
RxC_PACKING_MISS_VN0	0x4E	0-2	VN0 message cant slot into flit
RxC_PACKING_MISS_VN1	0x4F	0-2	VN1 message cant slot into flit
RxC_HELD	0x50	0-2	Message Held
RxC_FLIT_GEN_HDR1	0x51	0-3	Flit Gen - Header 1
RxC_FLIT_GEN_HDR2	0x52	0-3	Flit Gen - Header 2
RxC_HDR_FLIT_NOT_SENT	0x53	0-3	Header Not Sent
RxC_HDR_FLITS_SENT	0x54	0-3	Sent Header Flit
RxC_DATA_FLITS_NOT_SENT	0x55	0-3	Data Flit Not Sent
RxC_FLITS_SLOT_BL	0x56	0-3	Slotting BL Message Into Header Flit
RxC_FLITS_GEN_BL	0x57	0-3	Generating BL Data Flit Sequence
RxC_FLITS_MISC	0x58	0-3	Slot 2 requests
RxC_VNA_CRD_MISC	0x59	0-3	Remote VNA credits
RxC_VNA_CRD	0x5A	0-3	Remote VNA Credits
VN0_CREDITS_USED	0x5B	0-3	VN0 Credit Used
VN1_CREDITS_USED	0x5C	0-3	VN1 Credit Used
VN0_NO_CREDITS	0x5D	0-3	VN0 No Credits
VN1_NO_CREDITS	0x5E	0-3	VN1 No Credits
RxC_CRD_MISC	0x5F	0-3	Miscellaneous Credit Events
RxC_CRD_OCC	0x60	0-3	Credit Occupancy
XPT_PFTCH	0x61	0-3	XPT Prefetch messages
WB_PENDING	0x7D	0-3	
WB_OCC_COMPARE	0x7E	0-3	

2.9.3 M3UPI Box Performance Monitor Event List

The section enumerates 5th Gen Intel® Xeon® Scalable Processor performance monitoring events for the M3UPI Box.

CHA_AD_CREDITS_EMPTY

- **Title:**
- **Category:** Egress Credit Events
- **Event Code:** 0x22
- **Register Restrictions :** 0-3
- **Definition:** No credits available to send to Cbox on the AD ring (covers higher CBoxes).

Table 2-368. Unit Masks for CHA_AD_CREDITS_EMPTY

Extension	umask [15:8]	Description
VNA	bxxxxxxx1	VNA Messages
WB	bxxxxxx1x	Write backs
REQ	bxxxxx1xx	Requests
SNP	bxxxx1xxx	Snoops

CLOCKTICKS

- **Title:**
- **Category:** Clock Events
- **Event Code:** 0x1
- **Register Restrictions :** 0-3
- **Definition:** Counts the number of UCLKs in the M3 UCLK domain. This could be slightly different than the count in the Ubox because of enable or freeze delays. However, because the M3 is close to the Ubox, they generally should not diverge by more than a handful of cycles.

D2C_SENT

- **Title:**
- **Category:** Special Egress Events
- **Event Code:** 0x2B
- **Register Restrictions :** 0-3
- **Definition:** Count cases BL sends direct to core.

D2U_SENT

- **Title:**
- **Category:** Special Egress Events
- **Event Code:** 0x2A
- **Register Restrictions :** 0-3
- **Definition:** Cases where SMI3 sends D2U command.

M2_BL_CREDITS_EMPTY

- **Title:**
- **Category:** Egress Credit Events
- **Event Code:** 0x23
- **Register Restrictions :** 0-3
- **Definition:** No vn0 and VNA credits available to send to M2.

Table 2-369. Unit Masks for M2_BL_CREDITS_EMPTY

Extension	umask [15:8]	Description
IIO1_NCB	bxxxxxxx1	IIO0 and IIO1 share the same ring destination. (1 VN0 credit only)
IIO2_NCB	bxxxxxx1x	IIO2
IIO3_NCB	bxxxxx1xx	IIO3
IIO4_NCB	bxxxx1xxx	IIO4
IIO5_NCB	bxxx1xxxx	IIO5
UBOX_NCB0	bxx1xxxxx	Ubox
NCS	bx1xxxxxx	
NCS_SEL	b1xxxxxxx	Selected M2p BL NCS credits

MULTI_SLOT_RCVD

- **Title:**
- **Category:** Special Egress Events
- **Event Code:** 0x3E
- **Register Restrictions :** 0-3
- **Definition:** Multi slot flit received - S0, S1, or S2 populated (can use AK S0/S1 masks for AK allocations).

Table 2-370. Unit Masks for MULTI_SLOT_RCVD

Extension	umask [15:8]	Description
AD_SLOT0	bxxxxxxx1	AD - Slot 0
AD_SLOT1	bxxxxxx1x	AD - Slot 1
AD_SLOT2	bxxxxx1xx	AD - Slot 2
BL_SLOT0	bxxxx1xxx	BL - Slot 0
AK_SLOT0	bxxx1xxxx	AK - Slot 0
AK_SLOT2	bxx1xxxxx	AK - Slot 2

RxC_ARB_LOST_VN0

- **Title:**
- **Category:** Ingress Arbitration Events
- **Event Code:** 0x4B
- **Register Restrictions :**

- **Definition:** VN0 message requested but lost arbitration.

Table 2-371. Unit Masks for RxC_ARB_LOST_VN0

Extension	umask [15:8]	Description
AD_REQ	bxxxxxxx1	REQ on AD Home (REQ) messages on AD. REQ is generally used to send requests, request responses, and snoop responses.
AD_SNP	bxxxxxx1x	SNP on AD Snoops (SNP) messages on AD. SNP is used for outgoing snoops.
AD_RSP	bxxxxx1xx	RSP on AD Response (RSP) messages on AD. RSP packets are used to transmit a variety of protocol flits including grants and completions (CMP).
BL_RSP	bxxxx1xxx	RSP on BL Response (RSP) messages on BL. RSP packets are used to transmit a variety of protocol flits including grants and completions (CMP).
BL_WB	bxxx1xxxx	WB on BL Data Response (WB) messages on BL. WB is generally used to transmit data with coherency. For example, remote reads and writes, or cache to cache transfers will transmit their data using WB.
BL_NCB	bxx1xxxxx	NCB on BL Non-Coherent Broadcast (NCB) messages on BL. NCB is generally used to transmit data without coherency. For example, non-coherent read data returns.
BL_NCS	bx1xxxxxx	NCS on BL Non-Coherent Standard (NCS) messages on BL.

RxC_ARB_LOST_VN1

- **Title:**
- **Category:** Ingress Arbitration Events
- **Event Code:** 0x4C
- **Register Restrictions :**
- **Definition:** VN1 message requested but lost arbitration.

Table 2-372. Unit Masks for RxC_ARB_LOST_VN1 (Sheet 1 of 2)

Extension	umask [15:8]	Description
AD_REQ	bxxxxxxx1	REQ on AD Home (REQ) messages on AD. REQ is generally used to send requests, request responses, and snoop responses.
AD_SNP	bxxxxxx1x	SNP on AD Snoops (SNP) messages on AD. SNP is used for outgoing snoops.
AD_RSP	bxxxxx1xx	RSP on AD Response (RSP) messages on AD. RSP packets are used to transmit a variety of protocol flits including grants and completions (CMP).
BL_RSP	bxxxx1xxx	RSP on BL Response (RSP) messages on BL. RSP packets are used to transmit a variety of protocol flits including grants and completions (CMP).
BL_WB	bxxx1xxxx	WB on BL Data Response (WB) messages on BL. WB is generally used to transmit data with coherency. For example, remote reads and writes, or cache to cache transfers will transmit their data using WB.

Table 2-372. Unit Masks for RxC_ARB_LOST_VN1 (Sheet 2 of 2)

Extension	umask [15:8]	Description
BL_NCB	bxx1xxxxx	NCB on BL Non-Coherent Broadcast (NCB) messages on BL. NCB is generally used to transmit data without coherency. For example, non-coherent read data returns.
BL_NCS	bx1xxxxxx	NCS on BL Non-Coherent Standard (NCS) messages on BL.

RxC_ARB_MISC

- **Title:**
- **Category:** Ingress Arbitration Events
- **Event Code:** 0x4D
- **Register Restrictions :**
- **Definition:**

Table 2-373. Unit Masks for RxC_ARB_MISC

Extension	umask [15:8]	Description
NO_PROG_AD_VN0	bxxxxxxx1	No Progress on Pending AD VN0 Arbitration stage made no progress on pending ad VN0 messages because slotting stage cannot accept new message
NO_PROG_AD_VN1	bxxxxxx1x	No Progress on Pending AD VN1 Arbitration stage made no progress on pending ad VN1 messages because slotting stage cannot accept new message
NO_PROG_BL_VN0	bxxxxx1xx	No Progress on Pending BL VN0 Arbitration stage made no progress on pending bl VN0 messages because slotting stage cannot accept new message
NO_PROG_BL_VN1	bxxxx1xxx	No Progress on Pending BL VN1 Arbitration stage made no progress on pending bl VN1 messages because slotting stage cannot accept new message
ADBL_PARALLEL_WIN_VN0	bxxx1xxxx	AD, BL Parallel Win VN0 AD and BL messages won arbitration concurrently / in parallel
ADBL_PARALLEL_WIN_VN1	bxx1xxxxx	AD, BL Parallel Win VN1 AD and BL messages won arbitration concurrently / in parallel
VN01_PARALLEL_WIN	bx1xxxxxx	VN0, VN1 Parallel Win VN0 and VN1 arbitration sub-pipelines had parallel winners (at least one AD or BL on each side)
ALL_PARALLEL_WIN	b1xxxxxxx	Max Parallel Win VN0 and VN1 arbitration sub-pipelines both produced AD and BL winners (maximum possible parallel winners)

RxC_ARB_NOCRD_VN0

- **Title:**
- **Category:** Ingress Arbitration Events
- **Event Code:** 0x47
- **Register Restrictions :**
- **Definition:** VN0 message is blocked from requesting arbitration due to lack of remote Intel UPI credits.

Table 2-374. Unit Masks for RxC_ARB_NOCRD_VN0

Extension	umask [15:8]	Description
AD_REQ	bxxxxxxx1	REQ on AD Home (REQ) messages on AD. REQ is generally used to send requests, request responses, and snoop responses.
AD_SNP	bxxxxxx1x	SNP on AD Snoops (SNP) messages on AD. SNP is used for outgoing snoops.
AD_RSP	bxxxxx1xx	RSP on AD Response (RSP) messages on AD. RSP packets are used to transmit a variety of protocol flits including grants and completions (CMP).
BL_RSP	bxxxx1xxx	RSP on BL Response (RSP) messages on BL. RSP packets are used to transmit a variety of protocol flits including grants and completions (CMP).
BL_WB	bxxx1xxxx	WB on BL Data Response (WB) messages on BL. WB is generally used to transmit data with coherency. For example, remote reads and writes, or cache to cache transfers will transmit their data using WB.
BL_NCB	bxx1xxxxx	NCB on BL Non-Coherent Broadcast (NCB) messages on BL. NCB is generally used to transmit data without coherency. For example, non-coherent read data returns.
BL_NCS	bx1xxxxxx	NCS on BL Non-Coherent Standard (NCS) messages on BL.

RxC_ARB_NOCRD_VN1

- **Title:**
- **Category:** Ingress Arbitration Events
- **Event Code:** 0x48
- **Register Restrictions :**
- **Definition:** VN1 message is blocked from requesting arbitration due to lack of remote Intel UPI credits.

Table 2-375. Unit Masks for RxC_ARB_NOCRD_VN1 (Sheet 1 of 2)

Extension	umask [15:8]	Description
AD_REQ	bxxxxxxx1	REQ on AD Home (REQ) messages on AD. REQ is generally used to send requests, request responses, and snoop responses.
AD_SNP	bxxxxxx1x	SNP on AD Snoops (SNP) messages on AD. SNP is used for outgoing snoops.
AD_RSP	bxxxxx1xx	RSP on AD Response (RSP) messages on AD. RSP packets are used to transmit a variety of protocol flits including grants and completions (CMP).
BL_RSP	bxxxx1xxx	RSP on BL Response (RSP) messages on BL. RSP packets are used to transmit a variety of protocol flits including grants and completions (CMP).
BL_WB	bxxx1xxxx	WB on BL Data Response (WB) messages on BL. WB is generally used to transmit data with coherency. For example, remote reads and writes, or cache to cache transfers will transmit their data using WB.

Table 2-375. Unit Masks for RxC_ARB_NOCRD_VN1 (Sheet 2 of 2)

Extension	umask [15:8]	Description
BL_NCB	bxx1xxxxx	NCB on BL Non-Coherent Broadcast (NCB) messages on BL. NCB is generally used to transmit data without coherency. For example, non-coherent read data returns.
BL_NCS	bx1xxxxxx	NCS on BL Non-Coherent Standard (NCS) messages on BL.

RxC_ARB_NOREQ_VN0

- **Title:**
- **Category:** Ingress Arbitration Events
- **Event Code:** 0x49
- **Register Restrictions :**
- **Definition:** VN0 message was not able to request arbitration while some other message won arbitration.

Table 2-376. Unit Masks for RxC_ARB_NOREQ_VN0

Extension	umask [15:8]	Description
AD_REQ	bxxxxxxx1	REQ on AD Home (REQ) messages on AD. REQ is generally used to send requests, request responses, and snoop responses.
AD_SNP	bxxxxxx1x	SNP on AD Snoops (SNP) messages on AD. SNP is used for outgoing snoops.
AD_RSP	bxxxxx1xx	RSP on AD Response (RSP) messages on AD. RSP packets are used to transmit a variety of protocol flits including grants and completions (CMP).
BL_RSP	bxxxx1xxx	RSP on BL Response (RSP) messages on BL. RSP packets are used to transmit a variety of protocol flits including grants and completions (CMP).
BL_WB	bxxx1xxxx	WB on BL Data Response (WB) messages on BL. WB is generally used to transmit data with coherency. For example, remote reads and writes, or cache to cache transfers will transmit their data using WB.
BL_NCB	bxx1xxxxx	NCB on BL Non-Coherent Broadcast (NCB) messages on BL. NCB is generally used to transmit data without coherency. For example, non-coherent read data returns.
BL_NCS	bx1xxxxxx	NCS on BL Non-Coherent Standard (NCS) messages on BL.

RxC_ARB_NOREQ_VN1

- **Title:**
- **Category:** Ingress Arbitration Events
- **Event Code:** 0x4A
- **Register Restrictions :**
- **Definition:** VN1 message was not able to request arbitration while some other message won arbitration.

Table 2-377. Unit Masks for RxC_ARB_NOREQ_VN1

Extension	umask [15:8]	Description
AD_REQ	bxxxxxxx1	REQ on AD Home (REQ) messages on AD. REQ is generally used to send requests, request responses, and snoop responses.
AD_SNP	bxxxxxx1x	SNP on AD Snoops (SNP) messages on AD. SNP is used for outgoing snoops.
AD_RSP	bxxxxx1xx	RSP on AD Response (RSP) messages on AD. RSP packets are used to transmit a variety of protocol flits including grants and completions (CMP).
BL_RSP	bxxxx1xxx	RSP on BL Response (RSP) messages on BL. RSP packets are used to transmit a variety of protocol flits including grants and completions (CMP).
BL_WB	bxxx1xxxx	WB on BL Data Response (WB) messages on BL. WB is generally used to transmit data with coherency. For example, remote reads and writes, or cache to cache transfers will transmit their data using WB.
BL_NCB	bxx1xxxxx	NCB on BL Non-Coherent Broadcast (NCB) messages on BL. NCB is generally used to transmit data without coherency.For example, non-coherent read data returns.
BL_NCS	bx1xxxxxx	NCS on BL Non-Coherent Standard (NCS) messages on BL.

RxC_BYPASSED

- **Title:**
- **Category:** Ingress Events
- **Event Code:** 0x40
- **Register Restrictions :** 0-2
- **Definition:** Number of times message is bypassed around the ingress queue.

Table 2-378. Unit Masks for RxC_BYPASSED

Extension	umask [15:8]	Description
AD_S0_IDLE	bxxxxxxx1	AD to Slot 0 on Idle AD is taking bypass to slot 0 of independent flit while pipeline is idle
AD_S0_BL_ARB	bxxxxxx1x	AD to Slot 0 on BL Arb AD is taking bypass to slot 0 of independent flit while bl message is in arbitration
AD_S1_BL_SLOT	bxxxxx1xx	AD + BL to Slot 1 AD is taking bypass to flit slot 1 while merging with bl message in same flit
AD_S2_BL_SLOT	bxxxx1xxx	AD + BL to Slot 2 AD is taking bypass to flit slot 2 while merging with bl message in same flit

RxC_CRD_MISC

- **Title:**
- **Category:** Ingress Credit Events
- **Event Code:** 0x5F
- **Register Restrictions :**
- **Definition:**

Table 2-379. Unit Masks for RxC_CRD_MISC

Extension	umask [15:8]	Description
ANY_BGF_FIFO	bxxxxxxx1	Any In BGF FIFO Indication that at least one packet (flit) is in the BGF (FIFO only)
ANY_BGF_PATH	bxxxxxx1x	Any in BGF Path Indication that at least one packet (flit) is in the BGF path (that is, pipe to FIFO)
VN0_NO_D2K_FOR_ARB	bxxxxx1xx	No D2K For Arb VN0 BL RSP message was blocked from arbitration request due to lack of D2K CMP credit
VN1_NO_D2K_FOR_ARB	bxxxx1xxx	VN1 BL RSP message was blocked from arbitration request due to lack of D2K CMP credits VN1 BL RSP message was blocked from arbitration request due to lack of D2K CMP credits
LT1_FOR_D2K	bxxx1xxxx	D2K credit count is less than 1 d2k credit count is less than 1
LT2_FOR_D2K	bxx1xxxxx	D2K credit count is less than 2 d2k credit count is less than 2

RxC_CRD_OCC

- **Title:**
- **Category:** Ingress Credit Events
- **Event Code:** 0x60
- **Register Restrictions :**
- **Definition:**

Table 2-380. Unit Masks for RxC_CRD_OCC

Extension	umask [15:8]	Description
VNA_IN_USE	bxxxxxxx1	VNA In Use Remote Intel UPI VNA credit occupancy (number of credits in use), accumulated across all cycles
FLITS_IN_FIFO	bxxxxxx1x	Packets in BGF FIFO Occupancy of M3UPI ingress -

RxC_CYCLES_NE_VN0

- **Title:**
- **Category:** Ingress Events
- **Event Code:** 0x43
- **Register Restrictions :**
- **Definition:** Counts the number of cycles when the Intel UPI ingress is not empty. This tracks one of the three rings that are used by the Intel UPI agent. This can be used in conjunction with the Intel UPI's *Ingress Occupancy Accumulator* event in order to calculate average queue occupancy. Multiple ingress buffers can be tracked at a given time using multiple counters.

Table 2-381. Unit Masks for RxC_CYCLES_NE_VN0

Extension	umask [15:8]	Description
AD_REQ	bxxxxxxx1	REQ on AD Home (REQ) messages on AD. REQ is generally used to send requests, request responses, and snoop responses.
AD_SNP	bxxxxxx1x	SNP on AD Snoops (SNP) messages on AD. SNP is used for outgoing snoops.
AD_RSP	bxxxxx1xx	RSP on AD Response (RSP) messages on AD. RSP packets are used to transmit a variety of protocol flits including grants and completions (CMP).
BL_RSP	bxxxx1xxx	RSP on BL Response (RSP) messages on BL. RSP packets are used to transmit a variety of protocol flits including grants and completions (CMP).
BL_WB	bxxx1xxxx	WB on BL Data Response (WB) messages on BL. WB is generally used to transmit data with coherency. For example, remote reads and writes, or cache to cache transfers will transmit their data using WB.
BL_NCB	bxx1xxxxx	NCB on BL Non-Coherent Broadcast (NCB) messages on BL. NCB is generally used to transmit data without coherency. For example, non-coherent read data returns.
BL_NCS	bx1xxxxxx	NCS on BL Non-Coherent Standard (NCS) messages on BL.

RxC_CYCLES_NE_VN1

- **Title:**
- **Category:** Ingress Events
- **Event Code:** 0x44
- **Register Restrictions :**
- **Definition:** Counts the number of allocations into the Intel UPI VN1 ingress. This tracks one of the three rings that are used by the Intel UPI agent. This can be used in conjunction with the Intel UPI *VN1 Ingress Occupancy Accumulator* event in order to calculate average queue latency. Multiple ingress buffers can be tracked at a given time using multiple counters.

Table 2-382. Unit Masks for RxC_CYCLES_NE_VN1 (Sheet 1 of 2)

Extension	umask [15:8]	Description
AD_REQ	bxxxxxxx1	REQ on AD Home (REQ) messages on AD. REQ is generally used to send requests, request responses, and snoop responses.
AD_SNP	bxxxxxx1x	SNP on AD Snoops (SNP) messages on AD. SNP is used for outgoing snoops.
AD_RSP	bxxxxx1xx	RSP on AD Response (RSP) messages on AD. RSP packets are used to transmit a variety of protocol flits including grants and completions (CMP).
BL_RSP	bxxxx1xxx	RSP on BL Response (RSP) messages on BL. RSP packets are used to transmit a variety of protocol flits including grants and completions (CMP).
BL_WB	bxxx1xxxx	WB on BL Data Response (WB) messages on BL. WB is generally used to transmit data with coherency. For example, remote reads and writes, or cache to cache transfers will transmit their data using WB.

Table 2-382. Unit Masks for RxC_CYCLES_NE_VN1 (Sheet 2 of 2)

Extension	umask [15:8]	Description
BL_NCB	bxx1xxxxx	NCB on BL Non-Coherent Broadcast (NCB) messages on BL. NCB is generally used to transmit data without coherency. For example, non-coherent read data returns.
BL_NCS	bx1xxxxxx	NCS on BL Non-Coherent Standard (NCS) messages on BL.

RxC_DATA_FLITS_NOT_SENT

- **Title:**
- **Category:** Ingress Flit Events
- **Event Code:** 0x55
- **Register Restrictions :**
- **Definition:** Data flit is ready for transmission but could not be sent.

Table 2-383. Unit Masks for RxC_DATA_FLITS_NOT_SENT

Extension	umask [15:8]	Description
ALL	bxxxxxxx1	All data flit is ready for transmission but could not be sent for any reason, for example, low credits, low TSV, stall injection
TSV_HI	bxxxxxx1x	TSV High data flit is ready for transmission but was not sent while TSV high
VALID_FOR_FLIT	bxxxxx1xx	Cycle valid for Flit data flit is ready for transmission but was not sent while cycle is valid for flit transmission
NO_BGF	bxxxx1xxx	No BGF Credits
NO_TXQ	bxxx1xxxx	No TxQ Credits

RxC_FLITS_GEN_BL

- **Title:**
- **Category:** Ingress Flit Events
- **Event Code:** 0x57
- **Register Restrictions :**
- **Definition:**

Table 2-384. Unit Masks for RxC_FLITS_GEN_BL (Sheet 1 of 2)

Extension	umask [15:8]	Description
P0_WAIT	bxxxxxxx1	Wait on Pump 0 generating bl data flit sequence; waiting for data pump 0
P1_WAIT	bxxxxxx1x	Wait on Pump 1 generating bl data flit sequence; waiting for data pump 1
P1P_TO_LIMBO	bxxxxx1xx	a bl message finished but is in limbo and moved to pump-1-pending logic
P1P_BUSY	bxxxx1xxx	pump-1-pending logic is tracking at least one message

Table 2-384. Unit Masks for RxC_FLITS_GEN_BL (Sheet 2 of 2)

Extension	umask [15:8]	Description
P1P_AT_LIMIT	bxxx1xxxx	pump-1-pending logic is at capacity (pending table plus completion FIFO at limit)
P1P_HOLD_P0	bxx1xxxxx	pump-1-pending logic is at or near capacity, such that pump-0-only bl messages are getting stalled in slotting stage
P1P_FIFO_FULL	bx1xxxxxx	pump-1-pending completion FIFO is full

RxC_FLITS_MISC

- **Title:**
- **Category:** Ingress Flit Events
- **Event Code:** 0x58
- **Register Restrictions :**
- **Definition:**

Table 2-385. Unit Masks for RxC_FLITS_MISC

Extension	umask [15:8]	Description
S2REQ_RECEIVED	bxxxxxxx1	slot 2 request received from link layer while idle (with no slot 2 request active immediately prior)
S2REQ_WITHDRAWN	bxxxxxx1x	slot 2 request withdrawn during hold-off period or service window
S2REQ_IN_HOLDOFF	bxxxxx1xx	slot 2 request naturally serviced during hold-off period
S2REQ_IN_SERVICE	bxxxx1xxx	slot 2 request forcibly serviced during service window

RxC_FLITS_SLOT_BL

- **Title:**
- **Category:** Ingress Flit Events
- **Event Code:** 0x56
- **Register Restrictions :**
- **Definition:**

Table 2-386. Unit Masks for RxC_FLITS_SLOT_BL (Sheet 1 of 2)

Extension	umask [15:8]	Description
ALL	bxxxxxxx1	All
NEED_DATA	bxxxxxx1x	Needs Data Flit BL message requires data flit sequence
P0_WAIT	bxxxxx1xx	Wait on Pump 0 Waiting for header pump 0
P1_WAIT	bxxxx1xxx	Wait on Pump 1 Waiting for header pump 1
P1_NOT_REQ	bxxx1xxxx	Do not Need Pump 1 Header pump 1 is not required for flit

Table 2-386. Unit Masks for RxC_FLITS_SLOT_BL (Sheet 2 of 2)

Extension	umask [15:8]	Description
P1_NOT_REQ_BUT_BUBBLE	bxx1xxxxx	Do not Need Pump 1 - Bubble Header pump 1 is not required for flit but flit transmission delayed
P1_NOT_REQ_NOT_AVAIL	bx1xxxxxx	Do not Need Pump 1 - Not Avail Header pump 1 is not required for flit and not available

RxC_FLIT_GEN_HDR1

- **Title:**
- **Category:** Ingress Flit Events
- **Event Code:** 0x51
- **Register Restrictions :**
- **Definition:** Events related to Header Flit Generation - Set 1.

Table 2-387. Unit Masks for RxC_FLIT_GEN_HDR1

Extension	umask [15:8]	Description
ACCUM	bxxxxxxx1	Accumulate Header flit slotting control state machine is in any accumulate state; multi-message flit may be assembled over multiple cycles
ACCUM_READ	bxxxxxx1x	Accumulate Ready header flit slotting control state machine is in accum_ready state; flit is ready to send but transmission is blocked; more messages may be slotted into flit
ACCUM_WASTED	bxxxxx1xx	Accumulate Wasted Flit is being assembled over multiple cycles, but no additional message is being slotted into flit in current cycle; accumulate cycle is wasted
AHEAD_BLOCKED	bxxxx1xxx	Run-Ahead - Blocked Header flit slotting entered run-ahead state; new header flit is started while transmission of prior, fully assembled flit is blocked
AHEAD_MSG1_DURING	bxxx1xxxx	Run-Ahead - Message run-ahead mode: one message slotted during run-ahead
AHEAD_MSG2_AFTER	bxx1xxxxx	run-ahead mode: second message slotted immediately after run-ahead; potential run-ahead success
AHEAD_MSG2_SENT	bx1xxxxxx	run-ahead mode: two (or three) message flit sent immediately after run-ahead; complete run-ahead success
AHEAD_MSG1_AFTER	b1xxxxxxx	run-ahead mode: message was slotted only after run-ahead was over; run-ahead mode definitely wasted

RxC_FLIT_GEN_HDR2

- **Title:**
- **Category:** Ingress Flit Events
- **Event Code:** 0x52
- **Register Restrictions :**
- **Definition:** Events related to Header Flit Generation - Set 2.

Table 2-388. Unit Masks for RxC_FLIT_GEN_HDR2

Extension	umask [15:8]	Description
RMSTALL	bxxxxxxx1	Rate-matching Stall Rate-matching stall injected
RMSTALL_NOMSG	bxxxxxx1x	Rate-matching Stall - No Message Rate matching stall injected, but no additional message slotted during stall cycle
PAR	bxxxxx1xx	Parallel Ok new header flit construction may proceed in parallel with data flit sequence
PAR_MSG	bxxxx1xxx	Parallel Message message is slotted into header flit in parallel with data flit sequence
PAR_FLIT	bxxx1xxxx	Parallel Flit Finished header flit finished assembly in parallel with data flit sequence

RxC_HDR_FLITS_SENT

- **Title:**
- **Category:** Ingress Flit Events
- **Event Code:** 0x54
- **Register Restrictions :**
- **Definition:**

Table 2-389. Unit Masks for RxC_HDR_FLITS_SENT

Extension	umask [15:8]	Description
1_MSG	bxxxxxxx1	One Message One message in flit; VNA or non-VNA flit
2_MSGS	bxxxxxx1x	Two Messages Two messages in flit; VNA flit
3_MSGS	bxxxxx1xx	Three Messages Three messages in flit; VNA flit
1_MSG_VNX	bxxxx1xxx	One Message in non-VNA One message in flit; non-VNA flit
SLOTS_1	bxxx1xxxx	One Slot Taken
SLOTS_2	bxx1xxxxx	Two Slots Taken
SLOTS_3	bx1xxxxxx	All Slots Taken

RxC_HDR_FLIT_NOT_SENT

- **Title:**
- **Category:** Ingress Flit Events
- **Event Code:** 0x53
- **Register Restrictions :**
- **Definition:** header flit is ready for transmission but could not be sent.

Table 2-390. Unit Masks for RxC_HDR_FLIT_NOT_SENT

Extension	umask [15:8]	Description
ALL	bxxxxxxx1	All header flit is ready for transmission but could not be sent for any reason, for example, no credits, low TSV, stall injection
TSV_HI	bxxxxxx1x	TSV High header flit is ready for transmission but was not sent while TSV high
VALID_FOR_FLIT	bxxxxx1xx	Cycle valid for Flit header flit is ready for transmission but was not sent while cycle is valid for flit transmission
NO_BGF_CRD	bxxxx1xxx	No BGF Credits No BGF credits available
NO_TXQ_CRD	bxxx1xxxx	No TxQ Credits No TxQ credits available
NO_BGF_NO_MSG	bxx1xxxxx	No BGF Credits + No Extra Message Slotted No BGF credits available; no additional message slotted into flit
NO_TXQ_NO_MSG	bx1xxxxxx	No TxQ Credits + No Extra Message Slotted No TxQ credits available; no additional message slotted into flit

RxC_HELD

- **Title:**
- **Category:** Ingress Slotting Events
- **Event Code:** 0x50
- **Register Restrictions :** 0-2
- **Definition:**

Table 2-391. Unit Masks for RxC_HELD

Extension	umask [15:8]	Description
VN0	bxxxxxxx1	VN0 VN0 message(s) that could not be slotted into last VN0 flit are held in slotting stage while processing vn1 flit
VN1	bxxxxxx1x	VN1 VN1 message(s) that could not be slotted into last VN1 flit are held in slotting stage while processing vn0 flit
PARALLEL_ATTEMPT	bxxxxx1xx	Parallel Attempt AD and bl messages attempted to slot into the same flit in parallel
PARALLEL_SUCCESS	bxxxx1xxx	Parallel Success AD and bl messages were actually slotted into the same flit in parallel
CANT_SLOT_AD	bxxx1xxxx	Cannot Slot AD some AD message could not be slotted (logical OR of all AD events under INGR_SLOT_CANT_MC_VN{0,1})
CANT_SLOT_BL	bxx1xxxxx	Cannot Slot BL some BL message could not be slotted (logical OR of all BL events under INGR_SLOT_CANT_MC_VN{0,1})

RxC_INSERTS_VN0

- **Title:**
- **Category:** Ingress Events
- **Event Code:** 0x41
- **Register Restrictions :**
- **Definition:** Counts the number of allocations into the Intel UPI ingress. This tracks one of the three rings that are used by the Intel UPI agent. This can be used in conjunction with the Intel UPI *Ingress Occupancy Accumulator* event in order to calculate average queue latency. Multiple ingress buffers can be tracked at a given time using multiple counters.

Table 2-392. Unit Masks for RxC_INSERTS_VN0

Extension	umask [15:8]	Description
AD_REQ	bxxxxxxx1	REQ on AD Home (REQ) messages on AD. REQ is generally used to send requests, request responses, and snoop responses.
AD_SNP	bxxxxxx1x	SNP on AD Snoops (SNP) messages on AD. SNP is used for outgoing snoops.
AD_RSP	bxxxxx1xx	RSP on AD Response (RSP) messages on AD. RSP packets are used to transmit a variety of protocol flits including grants and completions (CMP).
BL_RSP	bxxxx1xxx	RSP on BL Response (RSP) messages on BL. RSP packets are used to transmit a variety of protocol flits including grants and completions (CMP).
BL_WB	bxxx1xxxx	WB on BL Data Response (WB) messages on BL. WB is generally used to transmit data with coherency. For example, remote reads and writes, or cache to cache transfers will transmit their data using WB.
BL_NCB	bxx1xxxxx	NCB on BL Non-Coherent Broadcast (NCB) messages on BL. NCB is generally used to transmit data without coherency. For example, non-coherent read data returns.
BL_NCS	bx1xxxxxx	NCS on BL Non-Coherent Standard (NCS) messages on BL.

RxC_INSERTS_VN1

- **Title:**
- **Category:** Ingress Events
- **Event Code:** 0x42
- **Register Restrictions :**
- **Definition:** Counts the number of allocations into the Intel UPI VN1 ingress. This tracks one of the three rings that are used by the Intel UPI agent. This can be used in conjunction with the Intel UPI VN1 *Ingress Occupancy Accumulator* event in order to calculate average queue latency. Multiple ingress buffers can be tracked at a given time using multiple counters.

Table 2-393. Unit Masks for RxC_INSERTS_VN1

Extension	umask [15:8]	Description
AD_REQ	bxxxxxxx1	REQ on AD Home (REQ) messages on AD. REQ is generally used to send requests, request responses, and snoop responses.
AD_SNP	bxxxxxx1x	SNP on AD Snoops (SNP) messages on AD. SNP is used for outgoing snoops.
AD_RSP	bxxxxx1xx	RSP on AD Response (RSP) messages on AD. RSP packets are used to transmit a variety of protocol flits including grants and completions (CMP).
BL_RSP	bxxxx1xxx	RSP on BL Response (RSP) messages on BL. RSP packets are used to transmit a variety of protocol flits including grants and completions (CMP).
BL_WB	bxxx1xxxx	WB on BL Data Response (WB) messages on BL. WB is generally used to transmit data with coherency. For example, remote reads and writes, or cache to cache transfers will transmit their data using WB.
BL_NCB	bxx1xxxxx	NCB on BL Non-Coherent Broadcast (NCB) messages on BL. NCB is generally used to transmit data without coherency. For example, non-coherent read data returns.
BL_NCS	bx1xxxxxx	NCS on BL Non-Coherent Standard (NCS) messages on BL.

RxC_OCCUPANCY_VN0

- **Title:**
- **Category:** Ingress Events
- **Event Code:** 0x45
- **Register Restrictions :**
- **Definition:** Accumulates the occupancy of a given Intel UPI VN1 ingress queue in each cycle. This tracks one of the three ring ingress buffers. This can be used with the Intel UPI *VN1 Ingress Not Empty* event to calculate average occupancy or the Intel UPI *VN1 Ingress Allocations* event in order to calculate average queuing latency.

Table 2-394. Unit Masks for RxC_OCCUPANCY_VN0 (Sheet 1 of 2)

Extension	umask [15:8]	Description
AD_REQ	bxxxxxxx1	REQ on AD Home (REQ) messages on AD. REQ is generally used to send requests, request responses, and snoop responses.
AD_SNP	bxxxxxx1x	SNP on AD Snoops (SNP) messages on AD. SNP is used for outgoing snoops.
AD_RSP	bxxxxx1xx	RSP on AD Response (RSP) messages on AD. RSP packets are used to transmit a variety of protocol flits including grants and completions (CMP).
BL_RSP	bxxxx1xxx	RSP on BL Response (RSP) messages on BL. RSP packets are used to transmit a variety of protocol flits including grants and completions (CMP).
BL_WB	bxxx1xxxx	WB on BL Data Response (WB) messages on BL. WB is generally used to transmit data with coherency. For example, remote reads and writes, or cache to cache transfers will transmit their data using WB.

Table 2-394. Unit Masks for RxC_OCCUPANCY_VN0 (Sheet 2 of 2)

Extension	umask [15:8]	Description
BL_NCB	bxx1xxxxx	NCB on BL Non-Coherent Broadcast (NCB) messages on BL. NCB is generally used to transmit data without coherency. For example, non-coherent read data returns.
BL_NCS	bx1xxxxxx	NCS on BL Non-Coherent Standard (NCS) messages on BL.

RxC_OCCUPANCY_VN1

- **Title:**
- **Category:** Ingress Events
- **Event Code:** 0x46
- **Register Restrictions :**
- **Definition:** Accumulates the occupancy of a given Intel UPI VN1 ingress queue in each cycle. This tracks one of the three ring ingress buffers. This can be used with the Intel UPI *VN1 Ingress Not Empty* event to calculate average occupancy or the Intel UPI *VN1 Ingress Allocations* event in order to calculate average queuing latency.

Table 2-395. Unit Masks for RxC_OCCUPANCY_VN1

Extension	umask [15:8]	Description
AD_REQ	bxxxxxxx1	REQ on AD Home (REQ) messages on AD. REQ is generally used to send requests, request responses, and snoop responses.
AD_SNP	bxxxxxx1x	SNP on AD Snoops (SNP) messages on AD. SNP is used for outgoing snoops.
AD_RSP	bxxxxx1xx	RSP on AD Response (RSP) messages on AD. RSP packets are used to transmit a variety of protocol flits including grants and completions (CMP).
BL_RSP	bxxxx1xxx	RSP on BL Response (RSP) messages on BL. RSP packets are used to transmit a variety of protocol flits including grants and completions (CMP).
BL_WB	bxxx1xxxx	WB on BL Data Response (WB) messages on BL. WB is generally used to transmit data with coherency. For example, remote reads and writes, or cache to cache transfers will transmit their data using WB.
BL_NCB	bxx1xxxxx	NCB on BL Non-Coherent Broadcast (NCB) messages on BL. NCB is generally used to transmit data without coherency. For example, non-coherent read data returns.
BL_NCS	bx1xxxxxx	NCS on BL Non-Coherent Standard (NCS) messages on BL.

RxC_PACKING_MISS_VN0

- **Title:**
- **Category:** Ingress Slotting Events
- **Event Code:** 0x4E
- **Register Restrictions :** 0-2
- **Definition:** Count cases where ingress has packets to send but did not have time to pack into flit before sending to agent so slot was left NULL which could have been used.

Table 2-396. Unit Masks for RxC_PACKING_MISS_VN0

Extension	umask [15:8]	Description
AD_REQ	bxxxxxxx1	REQ on AD Home (REQ) messages on AD. REQ is generally used to send requests, request responses, and snoop responses.
AD_SNP	bxxxxxx1x	SNP on AD Snoops (SNP) messages on AD. SNP is used for outgoing snoops.
AD_RSP	bxxxxx1xx	RSP on AD Response (RSP) messages on AD. RSP packets are used to transmit a variety of protocol flits including grants and completions (CMP).
BL_RSP	bxxxx1xxx	RSP on BL Response (RSP) messages on BL. RSP packets are used to transmit a variety of protocol flits including grants and completions (CMP).
BL_WB	bxxx1xxxx	WB on BL Data Response (WB) messages on BL. WB is generally used to transmit data with coherency. For example, remote reads and writes, or cache to cache transfers will transmit their data using WB.
BL_NCB	bxx1xxxxx	NCB on BL Non-Coherent Broadcast (NCB) messages on BL. NCB is generally used to transmit data without coherency. For example, non-coherent read data returns.
BL_NCS	bx1xxxxxx	NCS on BL Non-Coherent Standard (NCS) messages on BL.

RxC_PACKING_MISS_VN1

- **Title:**
- **Category:** Ingress Slotting Events
- **Event Code:** 0x4F
- **Register Restrictions :** 0-2
- **Definition:** Count cases where ingress has packets to send but did not have time to pack into flit before sending to agent so slot was left NULL which could have been used.

Table 2-397. Unit Masks for RxC_PACKING_MISS_VN1 (Sheet 1 of 2)

Extension	umask [15:8]	Description
AD_REQ	bxxxxxxx1	REQ on AD Home (REQ) messages on AD. REQ is generally used to send requests, request responses, and snoop responses.
AD_SNP	bxxxxxx1x	SNP on AD Snoops (SNP) messages on AD. SNP is used for outgoing snoops.
AD_RSP	bxxxxx1xx	RSP on AD Response (RSP) messages on AD. RSP packets are used to transmit a variety of protocol flits including grants and completions (CMP).
BL_RSP	bxxxx1xxx	RSP on BL Response (RSP) messages on BL. RSP packets are used to transmit a variety of protocol flits including grants and completions (CMP).
BL_WB	bxxx1xxxx	WB on BL Data Response (WB) messages on BL. WB is generally used to transmit data with coherency. For example, remote reads and writes, or cache to cache transfers will transmit their data using WB.

Table 2-397. Unit Masks for RxC_PACKING_MISS_VN1 (Sheet 2 of 2)

Extension	umask [15:8]	Description
BL_NCB	bxx1xxxxx	NCB on BL Non-Coherent Broadcast (NCB) messages on BL. NCB is generally used to transmit data without coherency. For example, non-coherent read data returns.
BL_NCS	bx1xxxxxx	NCS on BL Non-Coherent Standard (NCS) messages on BL.

RxC_VNA_CRD

- **Title:**
- **Category:** Ingress Credit Events
- **Event Code:** 0x5A
- **Register Restrictions :**
- **Definition:**

Table 2-398. Unit Masks for RxC_VNA_CRD

Extension	umask [15:8]	Description
CORRECTED	bxxxxxxx1	Corrected Number of remote VNA credits corrected (local return) per cycle
LT1	bxxxxxx1x	Level < 1 Remote VNA credit level is less than 1 (that is, no VNA credits available)
LT4	bxxxxx1xx	Level < 4 Remote VNA credit level is less than 4; bl (or ad requiring 4 VNA) cannot ARB on VNA
LT5	bxxxx1xxx	Level < 5 Remote VNA credit level is less than 5; parallel AD/BL ARB on VNA not possible
LT10	bxxx1xxxx	Level < 10 remote VNA credit level is less than 10; parallel VN0/VN1 ARB not possible
ANY_IN_USE	bxx1xxxxx	Any In Use At least one remote VNA credit is in use

RxC_VNA_CRD_MISC

- **Title:**
- **Category:** Ingress Credit Events
- **Event Code:** 0x59
- **Register Restrictions :**
- **Definition:**

Table 2-399. Unit Masks for RxC_VNA_CRD_MISC

Extension	umask [15:8]	Description
REQ_VN01_ALLOC_LT10	bxxxxxxx1	Remote VNA credit count was less than 10 and allocation to VN0 or VN1 was required
REQ_ADBL_ALLOC_L5	bxxxxxx1x	Remote VNA credit count was less than five and allocation to AD or BL messages was required
VN0_ONLY	bxxxxx1xx	Remote VNA credits were allocated only to VN0, not to VN1
VN1_ONLY	bxxxx1xxx	Remote VNA credits were allocated only to VN1, not to VN0
VN0_JUST_AD	bxxx1xxxx	On VN0, remote VNA credits were allocated only to AD messages, not to BL
VN0_JUST_BL	bxx1xxxxx	On VN0, remote VNA credits were allocated only to BL messages, not to AD
VN1_JUST_AD	bx1xxxxxx	On VN1, remote VNA credits were allocated only to AD messages, not to BL
VN1_JUST_BL	b1xxxxxxx	On VN1, remote VNA credits were allocated only to BL messages, not to AD

TxC_AD_ARB_FAIL

- **Title:**
- **Category:** ARB Events
- **Event Code:** 0x30
- **Register Restrictions :**
- **Definition:** AD ARB but no win; ARB request asserted but not won.

Table 2-400. Unit Masks for TxC_AD_ARB_FAIL

Extension	umask [15:8]	Description
VN0_REQ	bxxxxxxx1	VN0 REQ Messages
VN0_SNP	bxxxxxx1x	VN0 SNP Messages
VN0_RSP	bxxxxx1xx	VN0 RSP Messages
VN0_WB	bxxxx1xxx	VN0 WB Messages
VN1_REQ	bxxx1xxxx	VN1 REQ Messages
VN1_SNP	bxx1xxxxx	VN1 SNP Messages
VN1_RSP	bx1xxxxxx	VN1 RSP Messages
VN1_WB	b1xxxxxxx	VN1 WB Messages

TxC_AD_FLQ_BYPASS

- **Title:**
- **Category:** Special Egress Events
- **Event Code:** 0x2C
- **Register Restrictions :** 0-3
- **Definition:** Counts cases when the AD FlowQ is bypassed (S0, S1, and S2 indicate which slot was bypassed with S0 having the highest priority and S2 the least).

Table 2-401. Unit Masks for TxC_AD_FLQ_BYPASS

Extension	umask [15:8]	Description
AD_SLOT0	bxxxxxxx1	
AD_SLOT1	bxxxxxx1x	
AD_SLOT2	bxxxxx1xx	
BL_EARLY_RSP	bxxxx1xxx	

TxC_AD_FLQ_CYCLES_NE

- **Title:**
- **Category:** FlowQ Events
- **Event Code:** 0x27
- **Register Restrictions :** 0-3
- **Definition:** Number of cycles the AD egress queue is *Not Empty*.

Table 2-402. Unit Masks for TxC_AD_FLQ_CYCLES_NE

Extension	umask [15:8]	Description
VN0_REQ	bxxxxxxx1	VN0 REQ Messages
VN0_SNP	bxxxxxx1x	VN0 SNP Messages
VN0_RSP	bxxxxx1xx	VN0 RSP Messages
VN0_WB	bxxxx1xxx	VN0 WB Messages
VN1_REQ	bxxx1xxxx	VN1 REQ Messages
VN1_SNP	bxx1xxxxx	VN1 SNP Messages
VN1_RSP	bx1xxxxxx	VN1 RSP Messages
VN1_WB	b1xxxxxxx	VN1 WB Messages

TxC_AD_FLQ_INSERTS

- **Title:**
- **Category:** FlowQ Events
- **Event Code:** 0x2D
- **Register Restrictions :** 0-3
- **Definition:** Counts the number of allocations into the QPI FlowQ. This can be used in conjunction with the QPI FlowQ *Occupancy Accumulator* event in order to calculate average queue latency. Only a single FlowQ queue can be tracked at any given time. It is not possible to filter based on direction or polarity.

Table 2-403. Unit Masks for TxC_AD_FLQ_INSERTS (Sheet 1 of 2)

Extension	umask [15:8]	Description
VN0_REQ	bxxxxxxx1	VN0 REQ Messages
VN0_SNP	bxxxxxx1x	VN0 SNP Messages
VN0_RSP	bxxxxx1xx	VN0 RSP Messages

Table 2-403. Unit Masks for TxC_AD_FLQ_INSERTS (Sheet 2 of 2)

Extension	umask [15:8]	Description
VN0_WB	bxxxx1xxx	VN0 WB Messages
VN1_REQ	bxxx1xxxx	VN1 REQ Messages
VN1_SNP	bxx1xxxxx	VN1 SNP Messages
VN1_RSP	bx1xxxxxx	VN1 RSP Messages

TxC_AD_FLQ_OCCUPANCY

- **Title:**
- **Category:** FlowQ Events
- **Event Code:** 0x1C
- **Register Restrictions :** 0
- **Definition:**

Table 2-404. Unit Masks for TxC_AD_FLQ_OCCUPANCY

Extension	umask [15:8]	Description
VN0_REQ	bxxxxxxx1	VN0 REQ Messages
VN0_SNP	bxxxxxx1x	VN0 SNP Messages
VN0_RSP	bxxxxx1xx	VN0 RSP Messages
VN0_WB	bxxxx1xxx	VN0 WB Messages
VN1_REQ	bxxx1xxxx	VN1 REQ Messages
VN1_SNP	bxx1xxxxx	VN1 SNP Messages
VN1_RSP	bx1xxxxxx	VN1 RSP Messages

TxC_AK_FLQ_INSERTS

- **Title:**
- **Category:** FlowQ Events
- **Event Code:** 0x2F
- **Register Restrictions :** 0-3
- **Definition:**

TxC_AK_FLQ_OCCUPANCY

- **Title:**
- **Category:** FlowQ Events
- **Event Code:** 0x1E
- **Register Restrictions :** 0
- **Definition:**

TxC_BL_ARB_FAIL

- **Title:**
- **Category:** ARB Events
- **Event Code:** 0x35
- **Register Restrictions :**
- **Definition:** BL ARB but no win; ARB request asserted but not won.

Table 2-405. Unit Masks for TxC_BL_ARB_FAIL

Extension	umask [15:8]	Description
VN0_RSP	bxxxxxxx1	VN0 RSP Messages
VN0_WB	bxxxxxx1x	VN0 WB Messages
VN0_NCB	bxxxxx1xx	VN0 NCB Messages
VN0_NCS	bxxxx1xxx	VN0 NCS Messages
VN1_RSP	bxxx1xxxx	VN1 RSP Messages
VN1_WB	bxx1xxxxx	VN1 WB Messages
VN1_NCB	bx1xxxxxx	VN1 NCS Messages
VN1_NCS	b1xxxxxxx	VN1 NCB Messages

TxC_BL_FLQ_CYCLES_NE

- **Title:**
- **Category:** FlowQ Events
- **Event Code:** 0x28
- **Register Restrictions :** 0-3
- **Definition:** Number of cycles the BL egress queue is *Not Empty*.

Table 2-406. Unit Masks for TxC_BL_FLQ_CYCLES_NE

Extension	umask [15:8]	Description
VN0_REQ	bxxxxxxx1	VN0 REQ Messages
VN0_SNP	bxxxxxx1x	VN0 SNP Messages
VN0_RSP	bxxxxx1xx	VN0 RSP Messages
VN0_WB	bxxxx1xxx	VN0 WB Messages
VN1_REQ	bxxx1xxxx	VN1 REQ Messages
VN1_SNP	bxx1xxxxx	VN1 SNP Messages
VN1_RSP	bx1xxxxxx	VN1 RSP Messages
VN1_WB	b1xxxxxxx	VN1 WB Messages

TxC_BL_FLQ_INSERTS

- **Title:**
- **Category:** FlowQ Events
- **Event Code:** 0x2E
- **Register Restrictions :** 0-3

- **Definition:** Counts the number of allocations into the QPI FlowQ. This can be used in conjunction with the QPI FlowQ *Occupancy Accumulator* event in order to calculate average queue latency. Only a single FlowQ queue can be tracked at any given time. It is not possible to filter based on direction or polarity.

Table 2-407. Unit Masks for TxC_BL_FLQ_INSERTS

Extension	umask [15:8]	Description
VN0_NCB	bxxxxxxx1	VN0 RSP Messages
VN0_NCS	bxxxxxx1x	VN0 WB Messages
VN0_WB	bxxxxx1xx	VN0 NCB Messages
VN0_RSP	bxxxx1xxx	VN0 NCS Messages
VN1_NCB	bxxx1xxxx	VN1 RSP Messages
VN1_NCS	bxx1xxxxx	VN1 WB Messages
VN1_WB	bx1xxxxxx	VN1_NCS Messages
VN1_RSP	b1xxxxxxx	VN1_NCB Messages

TxC_BL_FLQ_OCCUPANCY

- **Title:**
- **Category:** FlowQ Events
- **Event Code:** 0x1D
- **Register Restrictions :** 0
- **Definition:**

Table 2-408. Unit Masks for TxC_BL_FLQ_OCCUPANCY

Extension	umask [15:8]	Description
VN0_RSP	bxxxxxxx1	VN0 RSP Messages
VN0_WB	bxxxxxx1x	VN0 WB Messages
VN0_NCB	bxxxxx1xx	VN0 NCB Messages
VN0_NCS	bxxxx1xxx	VN0 NCS Messages
VN1_RSP	bxxx1xxxx	VN1 RSP Messages
VN1_WB	bxx1xxxxx	VN1 WB Messages
VN1_NCB	bx1xxxxxx	VN1_NCS Messages
VN1_NCS	b1xxxxxxx	VN1_NCB Messages

TxC_BL_WB_FLQ_OCCUPANCY

- **Title:**
- **Category:** FlowQ Events
- **Event Code:** 0x1F
- **Register Restrictions :** 0
- **Definition:**

Table 2-409. Unit Masks for TxC_BL_WB_FLQ_OCCUPANCY

Extension	umask [15:8]	Description
VN0_LOCAL	b00000001	VN0 RSP Messages
VN0_THROUGH	b00000010	VN0 WB Messages
VN0_WRPULL	b00000100	VN0 NCB Messages
VN1_LOCAL	b00010000	VN1 RSP Messages
VN1_THROUGH	b00100000	VN1 WB Messages
VN1_WRPULL	b01000000	VN1_NCS Messages

UPI_PEER_AD_CREDITS_EMPTY

- **Title:**
- **Category:** Egress Credit Events
- **Event Code:** 0x20
- **Register Restrictions :** 0-3
- **Definition:** No credits available to send to Intel UPI on the AD ring.

Table 2-410. Unit Masks for UPI_PEER_AD_CREDITS_EMPTY

Extension	umask [15:8]	Description
VNA	bxxxxxxx1	VNA
VN0_REQ	bxxxxxx1x	VN0 REQ Messages
VN0_SNP	bxxxxx1xx	VN0 SNP Messages
VN0_RSP	bxxxx1xxx	VN0 RSP Messages
VN1_REQ	bxxx1xxxx	VN1 REQ Messages
VN1_SNP	bxx1xxxxx	VN1 SNP Messages
VN1_RSP	bx1xxxxxx	VN1 RSP Messages

UPI_PEER_BL_CREDITS_EMPTY

- **Title:**
- **Category:** Egress Credit Events
- **Event Code:** 0x21
- **Register Restrictions :**
- **Definition:** No credits available to send to Intel UPI on the BL ring (different between non-SMI and SMI mode).

Table 2-411. Unit Masks for UPI_PEER_BL_CREDITS_EMPTY (Sheet 1 of 2)

Extension	umask [15:8]	Description
VNA	bxxxxxxx1	VNA
VN0_RSP	bxxxxxx1x	VN0 REQ Messages
VN0_NCS_NCB	bxxxxx1xx	VN0 RSP Messages
VN0_WB	bxxxx1xxx	VN0 SNP Messages
VN1_RSP	bxxx1xxxx	VN1 REQ Messages

Table 2-411. Unit Masks for UPI_PEER_BL_CREDITS_EMPTY (Sheet 2 of 2)

Extension	umask [15:8]	Description
VN1_NCS_NCB	bxx1xxxxx	VN1 RSP Messages
VN1_WB	bx1xxxxxx	VN1 SNP Messages

UPI_PREFETCH_SPAWN

- **Title:**
- **Category:** Special Egress Events
- **Event Code:** 0x29
- **Register Restrictions :** 0-3
- **Definition:** Count cases where FlowQ causes spawn of prefetch to iMC/SMI3 target.

VN0_CREDITS_USED

- **Title:**
- **Category:** Link VN Credit Events
- **Event Code:** 0x5B
- **Register Restrictions :**
- **Definition:** Number of times a VN0 credit was used on the DRS message channel. In order for a request to be transferred across Intel UPI, it must be guaranteed to have a flit buffer on the remote socket to sink into. There are two credit pools, VNA and VN0. VNA is a shared pool used to achieve high performance. The VN0 pool has reserved entries for each message class and is used to prevent deadlock. Requests first attempt to acquire a VNA credit, and then fall back to VN0 if they fail. This counts the number of times a VN0 credit was used. Note that a single VN0 credit holds access to potentially multiple flit buffers. For example, a transfer that uses VNA could use nine flit buffers, and in that case it uses nine credits. A transfer on VN0 will only count a single credit even though it may use multiple buffers.

Table 2-412. Unit Masks for VN0_CREDITS_USED

Extension	umask [15:8]	Description
REQ	bxxxxxxx1	REQ on AD Home (REQ) messages on AD. REQ is generally used to send requests, request responses, and snoop responses.
SNP	bxxxxxx1x	SNP on AD Snoops (SNP) messages on AD. SNP is used for outgoing snoops.
RSP	bxxxxx1xx	RSP on AD Response (RSP) messages on AD. RSP packets are used to transmit a variety of protocol flits including grants and completions (CMP).
WB	bxxxx1xxx	RSP on BL Response (RSP) messages on BL. RSP packets are used to transmit a variety of protocol flits including grants and completions (CMP).
NCB	bxxx1xxxx	WB on BL Data Response (WB) messages on BL. WB is generally used to transmit data with coherency. For example, remote reads and writes, or cache to cache transfers will transmit their data using WB.
NCS	bxx1xxxxx	NCB on BL Non-Coherent Broadcast (NCB) messages on BL. NCB is generally used to transmit data without coherency. For example, non-coherent read data returns.

VN0_NO_CREDITS

- **Title:**
- **Category:** Link VN Credit Events
- **Event Code:** 0x5D
- **Register Restrictions :**
- **Definition:** Number of cycles there were no VN0 credits.

Table 2-413. Unit Masks for VN0_NO_CREDITS

Extension	umask [15:8]	Description
REQ	bxxxxxxx1	REQ on AD Home (REQ) messages on AD. REQ is generally used to send requests, request responses, and snoop responses.
SNP	bxxxxxx1x	SNP on AD Snoops (SNP) messages on AD. SNP is used for outgoing snoops.
RSP	bxxxxx1xx	RSP on AD Response (RSP) messages on AD. RSP packets are used to transmit a variety of protocol flits including grants and completions (CMP).
WB	bxxxx1xxx	RSP on BL Response (RSP) messages on BL. RSP packets are used to transmit a variety of protocol flits including grants and completions (CMP).
NCB	bxxx1xxxx	WB on BL Data Response (WB) messages on BL. WB is generally used to transmit data with coherency. For example, remote reads and writes, or cache to cache transfers will transmit their data using WB.
NCS	bxx1xxxxx	NCB on BL Non-Coherent Broadcast (NCB) messages on BL. NCB is generally used to transmit data without coherency. For example, non-coherent read data returns.

VN1_CREDITS_USED

- **Title:**
- **Category:** Link VN Credit Events
- **Event Code:** 0x5C
- **Register Restrictions :**
- **Definition:** Number of times a VN1 credit was used on the WB message channel. In order for a request to be transferred across QPI, it must be guaranteed to have a flit buffer on the remote socket to sink into. There are two credit pools, VNA and VN1. VNA is a shared pool used to achieve high performance. The VN1 pool has reserved entries for each message class and is used to prevent deadlock. Requests first attempt to acquire a VNA credit, and then fall back to VN1 if they fail. This counts the number of times a VN1 credit was used. Note that a single VN1 credit holds access to potentially multiple flit buffers. For example, a transfer that uses VNA could use nine flit buffers, and in that case it uses nine credits. A transfer on VN1 will only count a single credit even though it may use multiple buffers.

Table 2-414. Unit Masks for VN1_CREDITS_USED

Extension	umask [15:8]	Description
REQ	bxxxxxxx1	REQ on AD Home (REQ) messages on AD. REQ is generally used to send requests, request responses, and snoop responses.
SNP	bxxxxxx1x	SNP on AD Snoops (SNP) messages on AD. SNP is used for outgoing snoops.
RSP	bxxxxx1xx	RSP on AD Response (RSP) messages on AD. RSP packets are used to transmit a variety of protocol flits including grants and completions (CMP).
WB	bxxxx1xxx	RSP on BL Response (RSP) messages on BL. RSP packets are used to transmit a variety of protocol flits including grants and completions (CMP).
NCB	bxxx1xxxx	WB on BL Data Response (WB) messages on BL. WB is generally used to transmit data with coherency. For example, remote reads and writes, or cache to cache transfers will transmit their data using WB.
NCS	bxx1xxxxx	NCB on BL Non-Coherent Broadcast (NCB) messages on BL. NCB is generally used to transmit data without coherency. For example, non-coherent read data returns.

VN1_NO_CREDITS

- **Title:**
- **Category:** Link VN Credit Events
- **Event Code:** 0x5E
- **Register Restrictions :**
- **Definition:** Number of cycles there were no VN1 credits.

Table 2-415. Unit Masks for VN1_NO_CREDITS

Extension	umask [15:8]	Description
REQ	bxxxxxxx1	REQ on AD Home (REQ) messages on AD. REQ is generally used to send requests, request responses, and snoop responses.
SNP	bxxxxxx1x	SNP on AD Snoops (SNP) messages on AD. SNP is used for outgoing snoops.
RSP	bxxxxx1xx	RSP on AD Response (RSP) messages on AD. RSP packets are used to transmit a variety of protocol flits including grants and completions (CMP).
WB	bxxxx1xxx	RSP on BL Response (RSP) messages on BL. RSP packets are used to transmit a variety of protocol flits including grants and completions (CMP).
NCB	bxxx1xxxx	WB on BL Data Response (WB) messages on BL. WB is generally used to transmit data with coherency. For example, remote reads and writes, or cache to cache transfers will transmit their data using WB.
NCS	bxx1xxxxx	NCB on BL Non-Coherent Broadcast (NCB) messages on BL. NCB is generally used to transmit data without coherency. For example, non-coherent read data returns.

WB_OCC_COMPARE

- **Title:**
- **Category:** Write back Events
- **Event Code:** 0x7E
- **Register Restrictions :**
- **Definition:**

Table 2-416. Unit Masks for WB_OCC_COMPARE

Extension	umask [15:8]	Description
RT_GT_LOCALDEST_VN0	b0xxxxxx1	
RT_EQ_LOCALDEST_VN0	b0xxxxx1x	
RT_LT_LOCALDEST_VN0	b0xxxx1xx	
RT_GT_LOCALDEST_VN1	b0xx1xxxx	
RT_EQ_LOCALDEST_VN1	b0x1xxxxx	
RT_LT_LOCALDEST_VN1	b01xxxxxx	
BOTHNONZERO_RT_GT_LOCALDEST_VN0	b1xxxxxx1	
BOTHNONZERO_RT_EQ_LOCALDEST_VN0	b1xxxxx1x	
BOTHNONZERO_RT_LT_LOCALDEST_VN0	b1xxxx1xx	
BOTHNONZERO_RT_GT_LOCALDEST_VN1	b1xx1xxxx	
BOTHNONZERO_RT_EQ_LOCALDEST_VN1	b1x1xxxxx	
BOTHNONZERO_RT_LT_LOCALDEST_VN1	b11xxxxxx	

WB_PENDING

- **Title:**
- **Category:** Write back Events
- **Event Code:** 0x7D
- **Register Restrictions :**
- **Definition:**

Table 2-417. Unit Masks for WB_PENDING

Extension	umask [15:8]	Description
LOCALDEST_VN0	bxxxxxxx1	
ROUTETHRU_VN0	bxxxxxx1x	
LOCAL_AND_RT_VN0	bxxxxx1xx	
WAITING4PULL_VN0	bxxxx1xxx	
LOCALDEST_VN1	bxxx1xxxx	
ROUTETHRU_VN1	bxx1xxxxx	
LOCAL_AND_RT_VN1	bx1xxxxxx	
WAITING4PULL_VN1	b1xxxxxxx	

XPT_PFTCH

- **Title:**
- **Category:** XPT Events
- **Event Code:** 0x61
- **Register Restrictions :**
- **Definition:**

Table 2-418. Unit Masks for XPT_PFTCH

Extension	umask [15:8]	Description
ARRIVED	bxxxxxxx1	XPT prefetch message arrived in ingress pipeline
BYPASS	bxxxxxx1x	XPT prefetch message took bypass path
ARB	bxxxxx1xx	XPT prefetch message is making arbitration request
LOST_ARB	bxxxx1xxx	XPT prefetch message lost arbitration
FLITTED	bxxx1xxxx	XPT prefetch message was slotted into flit (non bypass)
LOST_QFULL	bx1xxxxx	XPT prefetch message was dropped because it was overwritten by new message while prefetch queue was full
LOST_OLD	bxx1xxxxx	XPT prefetch message was dropped because it became too old

2.10 Power Control (PCU) Performance Monitoring

The PCU is the primary power controller for the processor die, responsible for distributing power to core and uncore components, and thermal management. It runs in firmware on an internal micro-controller and coordinates the socket's power states.

The PCU algorithmically governs the P-state of the processor, C-state of the core and the package C-state of the socket. It enables the core to go to a higher performance state ("turbo mode") when the proper set of conditions are met. Conversely, the PCU will throttle the processor to a lower performance state when a thermal violation occurs.

Through specific events, the OS and the PCU will either promote or demote the C-State of each core by altering the voltage and frequency. The system power state (S-state) of all the sockets in the system is managed by the server legacy bridge in coordination with all socket PCUs.

The PCU communicates to all the other units through multiple PM link interfaces on-die and message channels to access their registers. The OS and BIOS communicates to the PCU through standardized MSR registers and ACPI.

Note: Power management is not completely centralized. Many units employ their own power saving features. Events that provide information about those features are captured in the PMON bocks of those units. For example, Intel® UPI link power saving states and memory CKE statistics are captured in the Intel® UPI PMON and IMC PMON respectively.

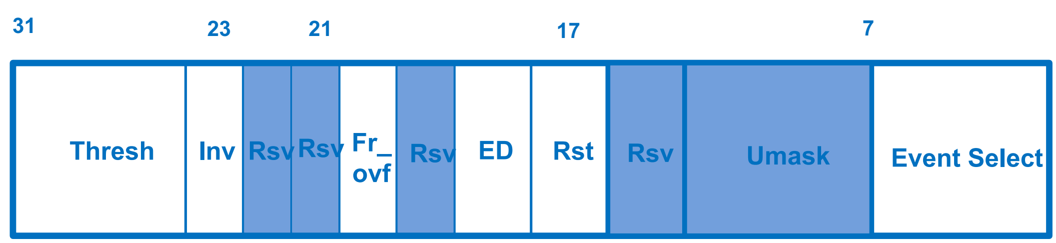
2.10.1 PCU Performance Monitoring Overview

The uncore PCU supports event monitoring through four 48-bit wide counters (PCU_MSR_PMON_CTR{3:0}). Each of these counters can be programmed (PCU_MSR_PMON_CTL{3:0}) to monitor any PCU event.

2.10.1.1 PCU PMON Counter Control - Difference from Baseline

The following table defines the difference in the layout of the PCU performance monitor control registers from the baseline presented in Chapter 1, “Introduction”.

Figure 2-8. PCU Counter Control Register for 5th Gen Intel® Xeon® Scalable Processor



2.10.2 Additional PCU Performance Monitoring

Context sensitive filtering is provided for through the PCU_MSR_PMON_BOX_FILTER register. Support for limited frequency/voltage band histogramming.

Each of the four bands provided for in the filter may be simultaneous tracked by the corresponding event because the use of the register as a filter is heavily overloaded, simultaneous application of this filter to additional events in the same run is severely limited.

Note: Address to the PCU specific filtering register can be found in Chapter 1. But there are some additional pieces of state of relevance to performance monitoring uses.

Table 2-419. PCU_MSR_PMON_BOX_FILTER Register – Field Definitions

Field	Bits	Attr	HW Reset Val	Description
rsv	63:48	RV	0	Reserved
filt31_24	31:24	RW-V	0	Band 3 - For Voltage/Frequency Band Event
filt23_16	23:16	RW-V	0	Band 2 - For Voltage/Frequency Band Event
filt15_8	15:8	RW-V	0	Band 1 - For Voltage/Frequency Band Event
filt7_0	7:0	RW-V	0	Band 0 - For Voltage/Frequency Band Event

2.10.3 PCU Performance Monitoring Events

The PCU provides the ability to capture information covering a wide range of the PCU’s functionality, including:

- Number of cores in a given C-state per-cycle.

- Core State Transitions - there are a larger number of events provided to track when cores transition C-state, when the enter or exit specific C-states, when they receive a C-state demotion, and so forth.
- Package State Transitions.
- Frequency/Voltage Banding - ability to measure the number of cycles the uncore was operating within a frequency or voltage 'band' that can be specified in a separate filter register.

Note: Given the nature of many of the PCU events, a great deal of additional information can be measured by setting the *.edge_det* bit. By doing so, an event such as "Cycles Changing Frequency" becomes "Number of Frequency Transitions".

On occupancy events:

Because it is not possible to "sync" the PCU occupancy counters by employing tricks such as bus lock before the events start incrementing, the PCU has provided fixed occupancy counters to track the major queues.

1. Cores in C0 (4 bits)
2. Cores in C6 (4 bits)

The PCU PMON implementation/programming is more complicated than many of the other units. As such, it is best to describe how to use them with a couple examples.

- Case 1: Cycles there was a voltage transition (simple event).
- Case 2: Cores in C0 (occupancy accumulation).
- Case 3: Cycles with more than four cores in C0 (occupancy thresholding).
- Case 4: Transitions into more than four cores in C0 (thresholding + edge detect).
- Case 5: Cycles a) with > four cores in C0 and b) there was a voltage transition.
- Case 6: Cycles a) with < four cores in C0 and b) frequency < 2.0 GHz.

Table 2-420. PCU Configuration Examples

	Case					
Config	**1**	**2**	**3**	**4**	**5**	**6**
Counter Control 0						
.ev_sel		0x80	0x80	0x80	0x80	0x80
.thresh		0x0	0x5	0x5	0x5	0x4
.invert		0	0	0	0	1
Counter Control 1						
.ev_sel	0x03				0x03	0x0B
Filter	0x00	0x00	0x00	0x00	0x00	0x14

2.10.4 PCU Box Events Ordered By Code

The following table summarizes the directly measured PCU box events.

Table 2-421. Directly Measured PCU Box Events (Sheet 1 of 2)

Symbol Name	Event Code	Ctrs	Description
NOTHING	0x0	0-3	Count Nothing
CLOCKTICKS	0x1	0-3	Clock ticks of the power control unit (PCU)
CROSSTHROT_LIGHT	0x11	0-3	Lightweight Cross Throttling Engaged
CROSSTHROT_HAMMER	0x12	0-3	Hammer Cross Throttling Engaged
THERMTHROT_MCP	0x14	0-3	MCP Thermal Throttling
THERMTHROT_IPM	0x15	0-3	IPM Thermal Throttling
THERMTHROT_UNCORE	0x16	0-3	Uncore Thermal Throttling
THERMTHROT_GT	0x18	0-3	GT Thermal Throttling
THERMTHROT_CORE	0x19	0-3	Core Thermal Throttling
HOT_MCP	0x1B	0-3	MCP is Hot
HOT_IPM	0x1C	0-3	IPM is Hot
HOT_UNCORE	0x1D	0-3	Uncore is Hot
HOT_GT	0x1E	0-3	GT is Hot
HOT_CORE	0x1F	0-3	Core is Hot
PKG_RESIDENCY_C0_CYCLES	0x2A	0-3	Package C State Residency - C0
PKG_RESIDENCY_C6_CYCLES	0x2D	0-3	Package C State Residency - C6
MEMORY_PHASE_SHEDDING_CYCLES	0x2F	0-3	Memory Phase Shedding Cycles
DEMOTIONS	0x30	0-3	
POWER_STATE_OCCUPANCY_CORES_C0	0x35	0-3	Number of cores in C0
POWER_STATE_OCCUPANCY_CORES_C6	0x37	0-3	Number of cores in C6
FREQ_MAX_LIMIT_THERMAL_CYCLES	0x4	0-3	Thermal Strongest Upper Limit Cycles
VR_HOT_CYCLES	0x42	0-3	VR Hot
FREQ_CLIP_AVX256	0x49	0-3	AVX256 Frequency Clipping
FREQ_CLIP_AVX512	0x4A	0-3	Intel® Advanced Vector Extensions 512 (Intel® AVX-512) Frequency Clipping
FREQ_MAX_POWER_CYCLES	0x5	0-3	Power Strongest Upper Limit Cycles
PMAX_THROTTLED_CYCLES	0x6		
CORE_TRANSITION_CYCLES	0x60	0-3	
TOTAL_TRANSITION_CYCLES	0x72	0-3	Total Core C State Transition Cycles
FREQ_MIN_IO_P_CYCLES	0x73	0-3	IO P Limit Strongest Lower Limit Cycles
FREQ_TRANS_CYCLES	0x74	0-3	Cycles spent changing Frequency
FIVR_PS_PS0_CYCLES	0x75	0-3	Phase Shed 0 Cycles
FIVR_PS_PS1_CYCLES	0x76	0-3	Phase Shed 1 Cycles
FIVR_PS_PS2_CYCLES	0x77	0-3	Phase Shed 2 Cycles

Table 2-421. Directly Measured PCU Box Events (Sheet 2 of 2)

Symbol Name	Event Code	Ctrs	Description
FIVR_PS_PS3_CYCLES	0x78	0-3	Phase Shed 3 Cycles
PROCHOT_INTERNAL_CYCLES	0x9	0-3	Internal PROCHOT
PROCHOT_EXTERNAL_CYCLES	0xA	0-3	External PROCHOT

2.10.5 PCU Box Common Metrics (Derived Events)

The following table summarizes metrics commonly calculated from PCU box events.

Table 2-422. Metrics Commonly Calculated From PCU Box Events

Symbol Name: Definition	Equation
PCT_CYC_FREQ_POWER_LTD: Percentage of Cycles the Max Frequency is limited by power	FREQ_MAX_POWER_CYCLES / CLOCKTICKS

2.10.6 PCU Box Performance Monitor Event List

The section enumerates 5th Gen Intel® Xeon® Scalable Processor performance monitoring events for the PCU box.

CLOCKTICKS

- **Title:**
- **Category:** PCLK Events
- **Event Code:** 0x1
- **Register Restrictions :** 0-3
- **Definition:** This event counts the number of PCLK cycles measured while the counter was enabled.

CORE_TRANSITION_CYCLES

- **Title:**
- **Category:** Core_C _State_Transition Events
- **Event Code:** 0x60
- **Register Restrictions :** 0-3
- **Definition:**

CROSSTHROT_HAMMER

- **Title:**
- **Category:** Thermal Throttling Events
- **Event Code:** 0x12
- **Register Restrictions :** 0-3
- **Definition:**

CROSSTHROT_LIGHT

- **Title:**
- **Category:** Thermal Throttling Events
- **Event Code:** 0x11
- **Register Restrictions :** 0-3
- **Definition:**

DEMOTIONS

- **Title:**
- **Category:** Core_C _State_Transition Events
- **Event Code:** 0x30
- **Register Restrictions :** 0-3
- **Definition:**

FIVR_PS_PS0_CYCLES

- **Title:**
- **Category:** FIVR Events
- **Event Code:** 0x75
- **Register Restrictions :** 0-3
- **Definition:** Cycles spent in phase-shedding power state 0.

FIVR_PS_PS1_CYCLES

- **Title:**
- **Category:** FIVR Events
- **Event Code:** 0x76
- **Register Restrictions :** 0-3
- **Definition:** Cycles spent in phase-shedding power state 1.

FIVR_PS_PS2_CYCLES

- **Title:**
- **Category:** FIVR Events
- **Event Code:** 0x77
- **Register Restrictions :** 0-3
- **Definition:** Cycles spent in phase-shedding power state 2.

FIVR_PS_PS3_CYCLES

- **Title:**
- **Category:** FIVR Events
- **Event Code:** 0x78
- **Register Restrictions :** 0-3
- **Definition:** Cycles spent in phase-shedding power state 3.

FREQ_CLIP_AVX256

- **Title:**
- **Category:** Frequency Clipping Events
- **Event Code:** 0x49
- **Register Restrictions :** 0-3
- **Definition:**

FREQ_CLIP_AVX512

- **Title:**
- **Category:** Frequency Clipping Events
- **Event Code:** 0x4A
- **Register Restrictions :** 0-3
- **Definition:**

FREQ_MAX_LIMIT_THERMAL_CYCLES

- **Title:**
- **Category:** Frequency Events
- **Event Code:** 0x4
- **Register Restrictions :** 0-3
- **Definition:** Number of cycles any frequency is reduced due to a thermal limit. Count only if throttling is occurring.

FREQ_MAX_POWER_CYCLES

- **Title:**
- **Category:** Frequency Events
- **Event Code:** 0x5
- **Register Restrictions :** 0-3
- **Definition:** Counts the number of cycles when power is the upper limit on frequency.

FREQ_MIN_IO_P_CYCLES

- **Title:**
- **Category:** Frequency Min Limit Events
- **Event Code:** 0x73
- **Register Restrictions :** 0-3
- **Definition:** Counts the number of cycles when IO P limit is preventing us from dropping the frequency lower. This algorithm monitors the needs to the IO subsystem on both local and remote sockets and will maintain a frequency high enough to maintain good IO BW. This is necessary for when all the IA cores on a socket are idle but a user still would like to maintain high IO bandwidth.

FREQ_TRANS_CYCLES

- **Title:**
- **Category:** FREQ_TRANS Events
- **Event Code:** 0x74
- **Register Restrictions :** 0-3
- **Definition:** Counts the number of cycles when the system is changing frequency. This can not be filtered by thread ID. One can also use it with the occupancy counter that monitors number of threads in C0 to estimate the performance impact that frequency transitions had on the system.

HOT_CORE

- **Title:**
- **Category:** Hot Events
- **Event Code:** 0x1F
- **Register Restrictions :** 0-3
- **Definition:** One of the cores is hot.

HOT_GT

- **Title:**
- **Category:** Hot Events
- **Event Code:** 0x1E
- **Register Restrictions :** 0-3
- **Definition:**

HOT_IPM

- **Title:**
- **Category:** Hot Events
- **Event Code:** 0x1C
- **Register Restrictions :** 0-3
- **Definition:** At least one in-Package Memory (iMC) is hot.

HOT_MCP

- **Title:**
- **Category:** Hot Events
- **Event Code:** 0x1B
- **Register Restrictions :** 0-3
- **Definition:** At least one MCP die is hot (except in-package memory).

HOT_UNCORE

- **Title:**
- **Category:** Hot Events
- **Event Code:** 0x1D
- **Register Restrictions :** 0-3
- **Definition:** Something in the Uncore is hot.

MEMORY_PHASE_SHEDDING_CYCLES

- **Title:**
- **Category:** MEMORY_PHASE_SHEDDING Events
- **Event Code:** 0x2F
- **Register Restrictions :** 0-3
- **Definition:** Counts the number of cycles that the PCU has triggered memory phase shedding. This is a mode that can be run in the iMC physicals that saves power at the expense of additional latency.

NOTHING

- **Title:**
- **Category:** PCLK Events
- **Event Code:** 0x0
- **Register Restrictions :** 0-3
- **Definition:** Equivalent to the SW telling the HW that it is not using the counter.

PKG_RESIDENCY_C0_CYCLES

- **Title:**
- **Category:** PKG_C_STATE_RESIDENCY Events
- **Event Code:** 0x2A
- **Register Restrictions :** 0-3
- **Definition:** Counts the number of cycles when the package was in C0. This event can be used in conjunction with edge detect to count C0 entrances (or exits using invert). Residency events do not include transition times.

PKG_RESIDENCY_C6_CYCLES

- **Title:**
- **Category:** PKG_C_STATE_RESIDENCY Events
- **Event Code:** 0x2D
- **Register Restrictions :** 0-3
- **Definition:** Counts the number of cycles when the package was in C6. This event can be used in conjunction with edge detect to count C6 entrances (or exits using invert). Residency events do not include transition times.

PMAX_THROTTLED_CYCLES

- **Title:**
- **Category:** Frequency Events
- **Event Code:** 0x6
- **Register Restrictions :**
- **Definition:**

POWER_STATE_OCCUPANCY_CORES_C0

- **Title:**
- **Category:** POWER_STATE_OCC Events
- **Event Code:** 0x35
- **Register Restrictions :** 0-3
- **Definition:** This is an occupancy event that tracks the number of cores that are in the chosen C-State. It can be used by itself to get the average number of cores in that C-state with thresholding to generate histograms, or with other PCU events and occupancy triggering to capture other details.

POWER_STATE_OCCUPANCY_CORES_C6

- **Title:**
- **Category:** POWER_STATE_OCC Events
- **Event Code:** 0x37
- **Register Restrictions :** 0-3
- **Definition:** This is an occupancy event that tracks the number of cores that are in the chosen C-State. It can be used by itself to get the average number of cores in

that C-state with thresholding to generate histograms, or with other PCU events and occupancy triggering to capture other details.

PROCHOT_EXTERNAL_CYCLES

- **Title:**
- **Category:** PROCHOT Events
- **Event Code:** 0xA
- **Register Restrictions :** 0-3
- **Definition:** Counts the number of cycles that we are in external PROCHOT mode. This mode is triggered when a sensor off the die determines that something off-die (like DRAM) is too hot and must throttle to avoid damaging the chip.

PROCHOT_INTERNAL_CYCLES

- **Title:**
- **Category:** PROCHOT Events
- **Event Code:** 0x9
- **Register Restrictions :** 0-3
- **Definition:** Counts the number of cycles that we are in internal PROCHOT mode. This mode is triggered when a sensor on the die determines that we are too hot and must throttle to avoid damaging the chip.

THERMTHROT_CORE

- **Title:**
- **Category:** Thermal Throttling Events
- **Event Code:** 0x19
- **Register Restrictions :** 0-3
- **Definition:** Thermal throttling of one of the core tiles by the PCU.

THERMTHROT_GT

- **Title:**
- **Category:** Thermal Throttling Events
- **Event Code:** 0x18
- **Register Restrictions :** 0-3
- **Definition:**

THERMTHROT_IPM

- **Title:**
- **Category:** Thermal Throttling Events
- **Event Code:** 0x15
- **Register Restrictions :** 0-3
- **Definition:** Thermal throttling of any iMC by the PCU.

THERMTHROT_MCP

- **Title:**
- **Category:** Thermal Throttling Events
- **Event Code:** 0x14
- **Register Restrictions :** 0-3
- **Definition:** Thermal throttling of any MCP by the PCU.

THERMTHROT_UNCORE

- **Title:**
- **Category:** Thermal Throttling Events
- **Event Code:** 0x16
- **Register Restrictions :** 0-3
- **Definition:**

TOTAL_TRANSITION_CYCLES

- **Title:**
- **Category:** Core_C _State_Transition Events
- **Event Code:** 0x72
- **Register Restrictions :** 0-3
- **Definition:** Number of cycles spent performing core C-state transitions across all cores.

VR_HOT_CYCLES

- **Title:**
- **Category:** VR_HOT Events
- **Event Code:** 0x42
- **Register Restrictions :** 0-3
- **Definition:** Number of cycles that a CPU SVID VR is hot. Does not cover DRAM VRs.

2.11 MDF Performance Monitoring

The MDF subsystem is a new IP built to support the new Intel® Xeon® architecture that bridges multiple dies with a embedded bridge system.

The MDF layers mesh protocol over the Embedded Multi-die Interconnect Bridge (EMIB).

Note: The EMIB is the physical layer and the MDF is the logical layer.

2.11.1 MDF Performance Monitoring Overview

Each MDF box supports event monitoring through four 48b wide counters (MDF_PMON_CTR/CTL{3:0}).

2.11.2 MDF Box Events Ordered By Code

The following table summarizes the directly measured MDF Box events.

Table 2-423. Directly Measured MDF Box Events

Symbol Name	Event Code	Ctrs	Description
Clockticks	0x1	0-3	
FAST_ASSERTED	0x15	0-3	Counts the number of cycles when the distress signals are asserted based on SBO Ingress threshold
GV_BLOCK	0x16	0-3	Counts the number of times the MSPMA sends a signal to SBO Ingress logic for blocking GV
GV_UNBLOCK	0x17	0-3	Counts the number of times the GV is unblocked.
RxR_BYPASS	0x14	0-3	Number of packets bypassing the SBO Ingress
RxR_FULL	0x11	0-3	Counts the number of cycles the Ingress buffers is full
RxR_INSERTS	0x12	0-3	Number of allocations into the SBO Ingress
RxR_OCCUPANCY	0x13	0-3	Occupancy counts for the SBO Ingress buffer

2.11.3 MDF Box Performance Monitor Event List

The section enumerates the 5th Gen Intel® Xeon® Scalable Processor performance monitoring events for the MDF box.

Clockticks

- **Title:**
- **Category:** Clockticks Event
- **Event Code:** 0x1
- **Register Restrictions :**
- **Definition:** Clockticks

FAST_ASSERTED

- **Title:**
- **Category:** SBO Events
- **Event Code:** 0x15
- **Register Restrictions :**
- **Definition:**

Table 2-424. Unit Masks for FAST_ASSERTED

Extension	umask [15:8]	Description
AD_BNC	bxxxxxxx1	AD bnc
BL_CRD	bxxxxxx1x	BL bnc

GV_BLOCK

- **Title:**
- **Category:** SBO Events
- **Event Code:** 0x16
- **Register Restrictions :**
- **Definition:**

GV_UNBLOCK

- **Title:**
- **Category:** SBO Events
- **Event Code:** 0x17
- **Register Restrictions :**
- **Definition:**

RxR_BYPASS

- **Title:**
- **Category:** SBO Events
- **Event Code:** 0x14
- **Register Restrictions :**
- **Definition:**

Table 2-425. Unit Masks for RxR_BYPASS

Extension	umask [15:8]	Description
AD_BNC	bxxxxxxx1	AD bnc
AD_CRD	bxxxxxx1x	AD crd
BL_BNC	bxxxxx1xx	BL bnc
BL_CRD	bxxxx1xxx	BL crd
AK	bxxx1xxxx	AK
IV	bxx1xxxxx	IV

RxR_FULL

- **Title:**
- **Category:** SBO Events
- **Event Code:** 0x11
- **Register Restrictions :**
- **Definition:**

Table 2-426. Unit Masks for RxR_FULL

Extension	umask [15:8]	Description
AD_CRD	bxxxxxxx1	AD
BL_CRD	bxxxxxx1x	BL
AK	bxxxxx1xx	AK
AKC	bxxxx1xxx	AKC
IV	bxxx1xxxx	IV

RxR_INSERTS

- **Title:**
- **Category:** SBO Events
- **Event Code:** 0x12
- **Register Restrictions :**
- **Definition:**

Table 2-427. Unit Masks for RxR_INSERTS

Extension	umask [15:8]	Description
AD_BNC	bxxxxxxx1	AD bnc
AD_CRD	bxxxxxx1x	AD crd
BL_BNC	bxxxxx1xx	BL bnc
BL_CRD	bxxxx1xxx	BL crd
AK	bxxx1xxxx	AK
IV	bxx1xxxxx	IV

RxR_OCCUPANCY

- **Title:**
- **Category:** SBO Events
- **Event Code:** 0x13
- **Register Restrictions :**
- **Definition:**

Table 2-428. Unit Masks for RxR_OCCUPANCY

Extension	umask [15:8]	Description
AD_BNC	bxxxxxxx1	AD bnc
AD_CRD	bxxxxxx1x	AD crd
BL_BNC	bxxxxx1xx	BL bnc
BL_CRD	bxxxx1xxx	BL crd
AK	bxxx1xxxx	AK
IV	bxx1xxxxx	IV

2.12 Compute Express Link* Performance Monitoring

The Compute Express Link* (CXL*) IP is responsible for transaction and link layer functionality associated with the transport of the CXL.cache and CXL.mem protocol over a physical link.

2.12.1 CXL Performance Monitoring Overview

Each CXL box supports 2 PMON (CXL CM/DP) blocks, unit 0 and unit 1, through eight 48b and four 48b wide counters. The reason we have two units is because CXL IP has two clock domains. The events for each of the domain are listed next and they cannot be used interchangeably.

Note: CXL CM Unit 1 - A PMON overflow message will not be sent out when the counters overflow and the UBOX does not send the freeze signal. The possible impact would be the counters increment as long as it reaches the maximum width of the counter and once it does the counters wraps to 0 and starts incrementing again.

2.12.2 CXL CM Box Events Ordered By Code

The following table summarizes the directly measured CXL CM box events.

Table 2-429. Directly Measured CXL CM Box Events (Sheet 1 of 2)

Symbol Name	Event Code	Ctrs	Description
CLOCKTICKS	0x1	0-7	Clock ticks
TxC_PACK_BUF_INSERTS	0x2	0-3	Number of allocations
RxC_MISC	0x40	4-7	
RxC_PACK_BUF_INSERTS	0x41	4-7	Number of allocations
RxC_PACK_BUF_NE	0x42	4-7	Number of cycles of not empty
RxC_AGF_INSERTS	0x43	4-7	Number of allocations

Table 2-429. Directly Measured CXL CM Box Events (Sheet 2 of 2)

Symbol Name	Event Code	Ctrs	Description
RxC_FLITS	0x4B	4-7	Counts the number of flits
TxC_FLITS	0x5	0-3	Counts the number of flits
RxC_PACK_BUF_FULL	0x52	4-7	Number of cycles the packing buffer is full

2.12.3 CXL CM Performance Monitor Event List

The section enumerates the 5th Gen Intel® Xeon® Scalable Processor performance monitoring events for the CXL CM box.

CLOCKTICKS

- **Title:**
- **Category:** Clock ticks event
- **Event Code:** 0x1
- **Register Restrictions :** 0-7
- **Definition:**

Table 2-430. Unit Masks for CLOCKTICKS

Extension	umask [15:8]	Description
Clockticks	bxxxxxxx1	

RxC_AGF_INSERTS

- **Title:**
- **Category:** Ingress AGF Events
- **Event Code:** 0x43
- **Register Restrictions :** 4-7
- **Definition:**

Table 2-431. Unit Masks for RxC_AGF_INSERTS (Sheet 1 of 2)

Extension	umask [15:8]	Description
CACHE_REQ0	bxxxxxxx1	Number of Allocation to Cache Req AGF0
CACHE_REQ1	bxxxxxx1x	Number of Allocation to Cache Rsp AGF
CACHE_RSP0	bxxxxx1xx	Number of Allocation to Cache Data AGF
CACHE_DATA	bxxxx1xxx	Number of Allocation to Mem Rxx AGF 0
MEM_REQ	bxxx1xxxx	Number of Allocation to Mem Data AGF

Table 2-431. Unit Masks for RxC_AGF_INSERTS (Sheet 2 of 2)

Extension	umask [15:8]	Description
MEM_DATA	bxx1xxxxx	Number of Allocation to Cache Req AGF 1
CACHE_RSP1	bx1xxxxxx	Number of Allocation to Cache Rsp AGF

RxC_FLITS

- **Title:**
- **Category:** Ingress Flit Events
- **Event Code:** 0x4B
- **Register Restrictions :** 4-7
- **Definition:**

Table 2-432. Unit Masks for RxC_FLITS

Extension	umask [15:8]	Description
VALID	bxxxxxxx1	Count the number of flits received
PROT	bxxxxxx1x	Count the number of protocol flits received
CTRL	bxxxxx1xx	Count the number of control flits received
NO_HDR	bxxxx1xxx	Count the number of header-less flits received
AK_HDR	bxxx1xxxx	Count the number of Flits with AK set
BE_HDR	bxx1xxxxx	Count the number of Flits with BE set
SZ_HDR	bx1xxxxxx	Count the number of Flits with SZ set
VALID_MSG	b1xxxxxxx	Count the number of valid messages in the flit

RxC_MISC

- **Title:**
- **Category:** Ingress Misc Events
- **Event Code:** 0x40
- **Register Restrictions :** 4-7
- **Definition:**

Table 2-433. Unit Masks for RxC_MISC (Sheet 1 of 2)

Extension	umask [15:8]	Description
LLCRD	bxxxxxxx1	Count the number of LLCRD flits sent
RETRY	bxxxxxx1x	Count the number of Retry flits sent

Table 2-433. Unit Masks for RxC_MISC (Sheet 2 of 2)

Extension	umask [15:8]	Description
INIT	bxxxxx1xx	Count the number of Init flits sent
CRC_ERRORS	bxxxx1xxx	Count the number of CRC errors detected

RxC_PACK_BUF_FULL

- **Title:**
- **Category:** Ingress Packing Buffer Events
- **Event Code:** 0x52
- **Register Restrictions :** 4-7
- **Definition:**

Table 2-434. Unit Masks for RxC_PACK_BUF_FULL

Extension	umask [15:8]	Description
CACHE_REQ	bxxxxxxx1	Number of cycles the Packing Buffer is Full
CACHE_RSP	bxxxxxx1x	Number of cycles the Packing Buffer is Full
CACHE_DATA	bxxxxx1xx	Number of cycles the Packing Buffer is Full
MEM_REQ	bxxxx1xxx	Number of cycles the Packing Buffer is Full
MEM_DATA	bxxx1xxxx	Number of cycles the Packing Buffer is Full

RxC_PACK_BUF_INSERTS

- **Title:**
- **Category:** Ingress Packing Buffer Events
- **Event Code:** 0x41
- **Register Restrictions :** 4-7
- **Definition:**

Table 2-435. Unit Masks for RxC_PACK_BUF_INSERTS

Extension	umask [15:8]	Description
CACHE_REQ	bxxxxxxx1	Number of Allocation to Cache Req Packing buffer
CACHE_RSP	bxxxxxx1x	Number of Allocation to Cache Rsp Packing buffer
CACHE_DATA	bxxxxx1xx	Number of Allocation to Cache Data Packing buffer
MEM_REQ	bxxxx1xxx	Number of Allocation to Mem Rxx Packing buffer
MEM_DATA	bxxx1xxxx	Number of Allocation to Mem Data Packing buffer

RxC_PACK_BUF_NE

- **Title:**
- **Category:** Ingress Packing Buffer Events
- **Event Code:** 0x42
- **Register Restrictions :** 4-7
- **Definition:**

Table 2-436. Unit Masks for RxC_PACK_BUF_NE

Extension	umask [15:8]	Description
CACHE_REQ	bxxxxxxx1	Number of cycles of Not Empty for Cache Req Packing buffer
CACHE_RSP	bxxxxxx1x	Number of cycles of Not Empty for Cache Rsp Packing buffer
CACHE_DATA	bxxxxx1xx	Number of cycles of Not Empty for Cache Data Packing buffer
MEM_REQ	bxxxx1xxx	Number of cycles of Not Empty for Mem Rxx Packing buffer
MEM_DATA	bxxx1xxxx	Number of cycles of Not Empty for Mem Data Packing buffer

TxC_PACK_BUF_INSERTS

- **Title:**
- **Category:** Egress Packing Buffer Events
- **Event Code:** 0x2
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-437. Unit Masks for TxC_PACK_BUF_INSERTS

Extension	umask [15:8]	Description
CACHE_REQ0	bxxxxxxx1	Number of Allocation to Cache Req Packing buffer
CACHE_RSP0	bxxxxxx1x	Number of Allocation to Cache Rsp0 Packing buffer
CACHE_DATA	bxxxxx1xx	Number of Allocation to Cache Data Packing buffer
MEM_REQ	bxxxx1xxx	Number of Allocation to Mem Rxx Packing buffer
MEM_DATA	bxxx1xxxx	Number of Allocation to Mem Data Packing buffer
CACHE_RSP1	bxx1xxxxx	Number of Allocation to Cache Req Packing buffer
CACHE_REQ1	bx1xxxxxx	Number of Allocation to Cache Rsp1 Packing buffer

TxC_FLITS

- **Title:**
- **Category:** Egress Flits Events
- **Event Code:** 0x5
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-438. Unit Masks for TxC_FLITS

Extension	umask [15:8]	Description
VALID	bxxxxxxx1	Counts the number of flits packed
PROT	bxxxxxx1x	Counts the number of protocol flits packed
CTRL	bxxxxx1xx	Counts the number of control flits packed
NO_HDR	bxxxx1xxx	Counts the number of header-less flits packed
AK_HDR	bxxx1xxxx	Counts the number of flits with AK set
BE_HDR	bxx1xxxxx	Counts the number of flits with BE set
SZ_HDR	bx1xxxxxx	Counts the number of flits with SZ set

2.12.4 CXL DP Box Events Ordered By Code

The following table summarizes the directly measured CXL DP box events.

Table 2-439. Directly Measured CXL DP Box Events

Symbol Name	Event Code	Ctrs	Description
CLOCKTICKS	0x1	0-3	Counts the number of UCLK ticks
TxC_AGF_INSERTS	0x2	0-3	

2.12.5 CXL DP Performance Monitor Event List

The section enumerates the 5th Gen Intel® Xeon® Scalable Processor performance monitoring events for the CXL DP box.

CLOCKTICKS

- **Title:**
- **Category:** Clock ticks Events
- **Event Code:** 0x1
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-440. Unit Masks for CLOCKTICKS

Extension	umask [15:8]	Description
Clockticks	bxxxxxxx1	

TxC_AGF_INSERTS

- **Title:**
- **Category:** Egress AGF Events
- **Event Code:** 0x2
- **Register Restrictions :** 0-3
- **Definition:**

Table 2-441. Unit Masks for TxC_AGF_INSERTS

Extension	umask [15:8]	Description
U2C_REQ	bxxxxxxx1	Number of Allocation to U2C Req AGF
U2C_RSP0	bxxxxxx1x	Number of Allocation to U2C Rsp AGF 0
U2C_RSP1	bxxxxx1xx	Number of Allocation to U2C Rsp AGF 1
U2C_DATA	bxxxx1xxx	Number of Allocation to U2C Data AGF
M2S_REQ	bxxx1xxxx	Number of Allocation to M2S Req AGF
M2S_DATA	bxx1xxxxx	Number of Allocation to M2S Data AGF

3 Reference for PMON Filtering

3.1 Packet Matching Reference(s)

3.1.1 Reference for CHA Packet Matching

In the CHA, the component responsible for managing the Last-level Cache (LLC) and maintaining coherency, the performance monitoring infrastructure allows a user to filter IDI packet traffic tracked in the TOR according to certain fields. The Message Class/ Opcode fields have been summarized in the following tables.

Note that the TOR is comprised of different logical queues managing different classes of requests. The Opcodes relevant to each logical queue class are presented in separate tables.

The following tables list the IDI opcodes, broken down by queue, that can be matched on with the above filter. The two Opcode match fields operate independently and the results are ORed together. It is not possible to measure events filtered by opcode that match in different fields.

IDI opcodes relevant to the Ingress Request Queue (IRQ).

Table 3-1. Opcode Match by IDI Packet Type (Relevant to IRQ) for Cn_MSR_PMON_BOX_FILTER.opc (Sheet 1 of 3)

opc Value	Opcode	Defn
0x100	RFO	Demand Data RFO - Full cache line read requests from agent for lines to be cached in any writable state
0x110	RFO_Pref	RFO Prefetch - Read for ownership request sent as prefetch from agent
0x101	CRd	Demand Code Read - Full cache-line read requests from core for lines to be cached in S, typically for code
0x105	CRd_UC	Uncacheable Code Read - Full cache-line read request from agent for lines not meant to be cached
0x111	CRd_Pref	Code Read Prefetch - Full cache-line read requests from core lines to be cached in S, typically from code. Treated as prefetch (that is, can be dropped)
0x102	DRd	Demand Data Read - Full cache-line read requests from core for lines to be cached in S or E, typically for data
0x112	DRd_Pref	Demand Data Read Prefetch - Full cache-line read requests from core for lines to be cached in S or E, typically for data. Treated as prefetch (that is, can be dropped)
0x104	DRd_Opt	Optimized Demand Data Read - Acts like DRd Prefetch except does not send LLC Miss response
0x114	DRd_Opt_Pref	Optimized Demand Data Read Prefetch - Acts like DRd except does not send LLC Miss response

Table 3-1. Opcode Match by IDI Packet Type (Relevant to IRQ) for Cn_MSR_PMON_BOX_FILTER.opc (Sheet 2 of 3)

opc Value	Opcode	Defn
0x106	DRdPTE	Demand Data Read for Page Walks - Full cache-line read requests from core for lines to be cached in S or E, for page walks
0x107	PRd	Partial Reads (UC) - Partial read requests of 0-32B (IIO can be up to 64B). Uncacheable.
0x10C	WCiLF	Streaming Store - Full - Write invalidate for full cache line of write combining stores
0x10D	WCiL	Streaming Store - Write invalidate for write combining stores
0x10E	UCRdF	Uncacheable Reads - Full - Full-line uncacheable read requests.
0x10F	WiL	Write Invalidate Line - Partial
0x118	CLFlush	Cacheline Flush - Invalidate cache line. All other agents cache lines must also be invalidated
0x11A	CLFlushOpt	Optimized Cacheline Flush - Invalidate cache line. All other agents cache lines must also be invalidated
0x11C	CLWB	Cacheline Flush WB - Invalidate cache line. All other agents cache lines must also be invalidated. Only writes modified data back to memory leaving cacheline in E if it was modified
0x11E	PCIRdCur	Read current - Read Current requests from IIO. Used to read data without changing state
0x13C	CLCleanse	Cacheline Cleanse - Only from IO
0x1A4	WbPushHint	 - Only from IO
0x184	WbMtoI	Request write back Modified invalidate line - Evict full M-state cache line from core. Guarantees core has no cached copies
0x185	WbMtoE	Request write back Modified set to Exclusive - Evict full M-state cache line from core
0x186	WbEFtoI	Request “clean” (E or F -state line) write back - Core guarantees it will no longer retain ownership of the line when the write-back completes
0x187	WbEFtoE	Request “clean” (E or F -state line) write back - Core may retain ownership of the line when the write-back completes
0x18C	WbStoI	Request write back. Shared to Invalidate - Clean line is being dropped. Allows snoop filter updates
0x188	ItoM	Request Invalidate Line - Request exclusive ownership of cache line. Agent guarantees entire cache line will be modified
0x18A	SpecItoM	Speculatively Request Invalidate Line - Request exclusive ownership of cache line. If speculation is correct, Agent guarantees entire cache line will be modified
0x198	LlcPrefRFO	LLC Prefetch RFO - Uncore will first look up the line in the LLC; for a cache hit, the LRU will be updated, on a miss, the RFO will be initiated

Table 3-1. Opcode Match by IDI Packet Type (Relevant to IRQ) for Cn_MSR_PMON_BOX_FILTER.opc (Sheet 3 of 3)

opc Value	Opcode	Defn
0x199	LlcPrefCode	LLC Prefetch Code - Uncore will first look up the line in the LLC; for a cache hit, the LRU will be updated, on a miss, the CRd will be initiated
0x19A	LlcPrefData	LLC Prefetch Data - Uncore will first look up the line in the LLC; for a cache hit, the LRU will be updated, on a miss, the DRd will be initiated
0x1D9	IntLog	Interrupt (Logically Addressed)
0x1DA	IntPhy	Interrupt (Physically Addressed)
0x1DB	IntPriUp	Interrupt Priority Update
0x1DE	SplitLock	Split Lock - Request to start split lock sequence
0x11D	FsRdCur	
0x13D	FsRdCurPtl	
0x109	PCommit	
0x14B	LLCWB	
0x1A4	WbPushHint	
0x1A5	WbPMPushHint	
0x180	CLDemote	
0x1DF	Lock	Lock - Request to start IDI lock sequence

IDI Opcodes relevant to the Ingress Subsequent Message Queue (ISMQ).

Table 3-2. Opcode Match by IDI Packet Type (Relevant to ISMQ) for Cn_MSR_PMON_BOX_FILTER.opc (Sheet 1 of 2)

opc Value	Opcode	Defn
0x000	RspI	Response I - Cache is in I
0x001	RspS	Response S - Cache is in S
0x026	RspV	Response V - Cache state unknown - For cases cache state does not need to be known
0x002	RspDataM	Response Data M
0x003	RspIFwdM	Response I Forward M
0x033	RspIFwdMPtl	Response I Forward M Partial - Only from IO
0x004	PullData	Pull Data
0x034	PullDataPtl	Pull Data Partial - Only from IO
0x005	PullDataBogus	Pull Data Bogus
0x006	Cmp	Completion - Only from Intel UPI

Table 3-2. Opcode Match by IDI Packet Type (Relevant to ISMQ) for Cn_MSR_PMON_BOX_FILTER.opc (Sheet 2 of 2)

opc Value	Opcode	Defn
0x007	CmpFwdCode	Completion Forward Code - Only from Intel UPI
0x008	CmpFwdInvItoE	Completion Forward Invalidate I to E - Only from Intel UPI
0x009	CmpPullData	Completion Pull Data - Only from Intel UPI
0x00B	CmpFwdInvOwn	Completion Forward Invalidate Own - Only from Intel UPI
0x00C	DataC_Cmp	Data Coherent Completion
0x01B	Victim	Victim - Only generated by CHA
0x01E	DataNc	Data Non Coherent - Only from Intel UPI
0x020	DataC	Data Complete - Only from Intel UPI
0x023	RspIFwdFE	Response S Forward F or E
0x024	RspSFwdFE	Response S Forward F or E
0x025	FwdCnflt	Forward Conflict
0x031	LLCVictim	LLC Victim - Only generated by CHA
0x035	MKTMEVictim	

IDI opcodes relevant to the Ingress Probe Queue (IPQ).

Table 3-3. Opcode Match by IDI Packet Type (Relevant to IPQ) for Cn_MSR_PMON_BOX_FILTER.opc

opc Value	Opcode	Defn
0x700	SnpCur	Snoop Current - Snoop to get uncacheable 'snapshot' of data
0x701	SnpCode	Snoop Code - Snoop requests from the uncore for lines intended to be cached in S at requester
0x702	SnpData	Snoop Data - Snoop requests from the uncore for lines intended to be cached in S or E state at the requester (the E state can be cached at requester if all cores respond with RspI)
0x703	SnpDataMig	Snoop Data Migratory - Snoop to get data in M, E, or S
0x704	SnpInvOwn	Snoop Invalidate Own - Snoop Invalidate Own - get data in M or E
0x705	SnpInvItoE	Snoop Invalidate - Snoop requests from the uncore for lines intended to be cached in E state at the requester

IDI opcodes relevant to the RRQ (Remote Request Queue).

Table 3-4. Opcode Match by IDI Packet Type (relevant to RRQ) for Cn_MSR_PMON_BOX_FILTER.opc

opc Value	Opcode	Defn
0x500	RdCur	Read Current - Request cache line in I. Typically issued by I/O proxy entities, RdCur is used to obtain a coherent snapshot of an uncached line
0x501	RdCode	Read Code - Read cache line in S
0x502	RdData	Read Data - Request cache line in either E or S. The choice between S and E is determined by whether or not per caching agent has cache line in S state
0x503	RdDataMig	Read Data Migratory - Same as RdData, except that peer cache can forward requested cache line in M state without any write back to memory
0x504	InvOwn	Read Invalidate Own - 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
0x505	InvXtoI	Invalidate X to I -
0x507	InvItoE	Invalidate I to E -
0x50C	RdInv	Read Invalidate - Request cache line in E from the home agent; any modified copy is committed to memory before receiving the data
0x50F	InvItoM	Invalidate I to M -

IDI opcodes relevant to the Write Back Queue (WBQ).

Table 3-5. Opcode Match by IDI Packet Type (Relevant to WBQ) for Cn_MSR_PMON_BOX_FILTER.opc (Sheet 1 of 2)

opc Value	Opcode	Defn
0x400	WbMtoI	Write back M to I - Evict full M-state cache line from core. Guarantees core has no cached copies. Write a cache line in M state back to memory and invalidate the line in the cache
0x401	WbMtoS	Write back M to S - Write a cache line in M state back to memory and transition its state to S
0x402	WbMtoE	Write back M to E - Evict full M-state cache line from core. Write a cache line in M state back to memory and transition its state to E
0x403	NonsnpWr	Non-Snoop Write - Write a line to memory
0x404	WbMtoIPtl	Write back M to I Partial - Write a cache line in M state back to memory, according to a byte-enable mask, and transition its state to I
0x406	WbMtoEPtl	Write back M to E Partial - Write a cache line in M state back to memory, according to a byte-enable mask, transition the line to E, and clear the line's mask in the cache

Table 3-5. Opcode Match by IDI Packet Type (Relevant to WBQ) for Cn_MSR_PMON_BOX_FILTER.opc (Sheet 2 of 2)

opc Value	Opcode	Defn
0x407	NonsnpWrPtl	Non-Snoop Write Partial - Write a line to memory according to byte-enable mask
0x408	WbPushMtoI	Write back Push M to I - Push cache line in M state to the HA; HA may push data to a local cache (in M state) or write the data to memory. Transition cache line to I
0x40B	WbFlush	Write back Flush - Hint for flushing writes in memory hierarchy. No data is sent with the request
0x40C	EvctCln	Evict Clean - Notification to home that a cache line in E state was invalidated in the cache
0x40D	NonSnpRd	Non-Snoop Read - Request a read only line (that is, an uncacheable 'snapshot') from memory

3.1.2 Reference for Intel UPI LL Packet Matching

In the Intel® UPI link layer, the component responsible for transmitting and receiving traffic crossing between sockets in a multi-socket machine, the performance monitoring infrastructure allows a user to filter Intel UPI packet traffic according to certain fields. A couple common fields, the Message Class/Opcode fields, have been summarized in the following tables.

Table 3-6. Intel® UPI Interconnect Packet Message Classes

Code	Name	Definition
b0000	REQ	Requests
b0001	SNP	Snoop
b0010	RSP - NoData	Non-Data Responses
b0011	---	
b0100	RSP - Data	Data Response
b0101	WB	Write Backs
b0110	NCB	Non-Coherent Bypass
b0111	NCS	Non-Coherent Standard

Table 3-7. UPI Opcode Match by Message Class (Sheet 1 of 2)

Opc	REQ	SNP	RSP2- NoData	
0000	RdCur	SnpCur	CmpU	
0001	RdCode	SnpCode	P2PCmpU	
0010	RdData	SnpData	RspI	
0011	RdDataMig	SnpDataMig	RspS	
0100	RdInvOwn	SnpInvOwn	RspFwd	
0101	InvXtoI	SnpInv	RspFwdI	
0110	InvItoM(*change)*	---	RspFwdS	

Table 3-7. UPI Opcode Match by Message Class (Sheet 2 of 2)

Opc	REQ	SNP	RSP2- NoData	
0111	InvItoE	---	---	
1000	---	SnpFCur	MirCmpU	
1001	---	SnpFCode	---	
1010	---	SnpFData	RspCnflt	
1011	---	SnpFDataMig	---	
1100	RdInv	SnpFInvOwn	CmpO	
1101	---	SnpFInv	FwdCnfltO	
1110	---	---	---	
1111	InvItoM	---	---	
Opc	**RSP4 - Data**	**WB**	**NCB**	**NCS**
0000	Data_M	WbMtoI	NcWr	NcRd
0001	Data_E	WbMtoS	WcWr	IntAck
0010	Data_SI	WbMtoE	---	---
0011	---	NonSnpWr	---	---
0100	Data_M_CmpO	WbMtoIPtl	---	NcRdPtl
0101	Data_E_CmpO	---	---	NcCfgRd
0110	Data_SI_CmpO	WbMtoEPtl	---	NcLTRd
0111	---	NonSnpWrPtl	---	NcIORd
1000	---	WbPushMtoI	NcMsgB	NcMsgS
1001	---	---	IntLogical	NcCfgWr
1010	RspFwdIWb	---	IntPhysical	NcLTWr
1011	RspFwdSWb	---(Change)	IntPrioUpd	NcIOWr
1100	RspIWb	EvctCln	NcWrPtl	---
1101	RspSWb	NonSnpRd	WcWrPtl	---
1110	---	---	---	---
1111	DebugData	---	NcP2PB	NcP2PS

Note: The Opcodes marked in *Italics* are not implemented in the 1st Gen Intel® Xeon® Scalable processor.

Table 3-8. UPI Opcodes (Alphabetical Listing) (Sheet 1 of 4)

Name	Opc	Msg Class	Gen By	Desc
CmpO	1100	RSP2		Completion message with no ordering requirements
CmpU	0000	RSP2		Completion message that must be ordered with forward responses.
DataE	0001	RSP4		Data in E
DataE_CmpO	0101	RSP4		Data in E with an ordered completion response
DataM	0000	RSP4		Data in M
DataM_CmpO	0100	RSP4		Data in M with an ordered completion response
DataSI	0010	RSP4		Depending on request, data in S or uncacheable 'snapshot' of data

Table 3-8. UPI Opcodes (Alphabetical Listing) (Sheet 2 of 4)

Name	Opc	Msg Class	Gen By	Desc
DataSI_CmpO	0110	RSP4		Depending on request, data in S or uncacheable ‘snapshot’ of data; with an ordered completion response
DebugData	1111	RSP4		Debug Data
EvctCln	1100	SNP		Notification to home that a cache line in E state was invalidated in the cache
FwdCnfltO	1101	WB		Ordered response from home agent to resolve conflict situation and let receiver properly process original snoop request. There is always a pre-allocated resource to sink the FwdCnfltO in the coherence agent
IntAck	0001	NCS		Interrupt acknowledge to legacy 8259 interrupt controller
IntLogical	1001	NCB		Logical mode interrupt to processor
IntPhysical	1010	NCB		Physical mode interrupt to processor
IntPrioUpd	1011	NCB		Interrupt priority update message to source interrupt agents
InvItoE	0111	REQ		Invalidate to E state. Requests exclusive ownership of a cache line without receiving data
InvItoM	1111	REQ		Invalidate to M state. Requests exclusive ownership of a cache line without receiving data and with the intent of performing a write back soon afterword
InvXtoI	0101	REQ		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). Requesting agent must invalidate the line in its cache before issuing this request
NcCfgRd	0101	NCS		Configuration read from configuration space
NcCfgWr	1001	NCS		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		Non-coherent Message (non-coherent bypass channel)
NcMsgS	1000	NCS		Non-coherent Message (Non-coherent standard channel)
NcP2PB	1111	NCB		Peer-to-peer transaction between I/O entities (non-coherent bypass channel)
NcP2PS	1111	NCS		Peer-to-peer transaction between I/O entities. (Non-coherent standard channel)
NcRd	0000	NCS		Read from non-coherent memory mapped I/O space
NcRdPtl	0100	NCS		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		Partial write to non-coherent memory mapped I/O space
NonSnpRd	1101	WB		Request a read only line (that is, an uncacheable ‘snapshot’) from memory
NonSnpWr	0011	WB		Write a line to memory
NonSnpWrPtl	0111	WB		Write a line to memory according to byte-enable mask

Table 3-8. UPI Opcodes (Alphabetical Listing) (Sheet 3 of 4)

Name	Opc	Msg Class	Gen By	Desc
P2PCmpU	0001	RSP2		Peer-to-peer completion message that must be ordered with forward responses
RdCode	0001	REQ		Read cache line in S
RdCur	0000	REQ		Request cache line in I. Typically issued by I/O proxy entities, RdCur is used to obtain a coherent snapshot of an uncached line
RdData	0010	REQ		Request cache line in either E or S. The choice between S and E is determined by whether or not per caching agent has cache line in S state
RdDataMig	0011	REQ		Same as RdData, except that peer cache can forward requested cache line in M state without any write back to memory
RdInv	1100	REQ		Request cache line in E from the home agent; any modified copy is committed to memory before receiving the data
RdInvOwn	0100	REQ		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	1010	RSP2		Peer has outstanding request to same address, is requesting an ordered forward response, and has allocated a resource for the forward
RspFwd	0100	RSP2		Copy of cache line was sent to requesting agent, cache state did not change
RspFwdI	0101	RSP2		Copy of cache line was sent to requesting agent, cache state was downgraded to I
RspFwdIWb	1010	RSP4		Modified line is being implicitly written back to memory, a copy of cache line was sent to requesting agent and the line was downgraded to I
RspFwdS	0110	RSP2		Copy of cache line was sent to requesting agent, cache state was downgraded to S
RspFwdSWb	1011	RSP4		Modified line is being implicitly written back to memory, a copy of cache line was sent to requesting agent and the line was downgraded to S
RspI	0010	RSP2		Cache is in I
RspIWb	1100	RSP4		Modified line is being implicitly written back to memory, cache line was downgraded to I
RspS	0011	RSP2		Cache is in S
RspSWb	1101	RSP4		Modified line is being implicitly written back to memory, cache line was downgraded to S
SnpCode	0001	SNP		Snoop Code - get data in S
SnpCur	0000	SNP		Snoop to get uncacheable 'snapshot' of data
SnpData	0010	SNP		Snoop Data - get data in E or S
SnpDataMig	0011	SNP		Snoop to get data in M, E, or S
SnpFCode	1001	SNP		Snoop Code - get data in S; Routing layer will handle distribution to all fanout peers
SnpFCur	1000	SNP		Snoop to get uncacheable 'snapshot' of data; Routing layer will handle distribution to all fanout peers
SnpFData	1010	SNP		Snoop Data - get data in E or S; Routing layer will handle distribution to all fanout peers

Table 3-8. UPI Opcodes (Alphabetical Listing) (Sheet 4 of 4)

Name	Opc	Msg Class	Gen By	Desc
SnpFDataMig	1011	SNP		Snoop to get data in M, E, or S; Routing layer will handle distribution to all fanout peers
SnpFInv	1101	SNP		Snoop to invalidate peer's cache, flushing any M copy to memory; Routing layer will handle distribution to all fanout peers
SnpFInvOwn	1100	SNP		Snoop Invalidate Own - get data in M or E; Routing layer will handle distribution to all fanout peers
SnpInv	0101	SNP		Snoop to invalidate peer's cache, flushing any M copy to memory
SnpInvOwn	0100	SNP		Snoop Invalidate Own - get data in M or E
SnpInvXtoI	1100	SNP		Snoop Invalidate Write Back M to I state. To invalidate peer caching agent, flushing any M state data to home
WBFlush	1011	WB		Hint for flushing writes in memory hierarchy. No data is sent with the request
WbMtoE	0010	WB		Write a cache line in M state back to memory and transition its state to E
WbMtoEPtl	0110	WB		Write a cache line in M state back to memory, according to a byte-enable mask, transition the line to E, and clear the line's mask in the cache
WbMtoI	0000	WB		Write a cache line in M state back to memory and invalidate the line in the cache
WbMtoIPtl	0100	WB		Write a cache line in M state back to memory, according to a byte-enable mask, and transition its state to I
WbMtoS	0001	WB		Write a cache line in M state back to memory and transition its state to S
WbPushMtoI	1000	WB		Push cache line in M state to the HA; HA may push data to a local cache (in M state) or write the data to memory. Transition cache line to I
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