POWER9 Performance Monitor Unit User's Guide

OpenPOWER

Version 1.2
28 November 2018

Printed in the United States of America November 2018

IBM Systems
294 Route 100, Building SOM4
Somers, NY 10589-3216

The IBM home page can be found at **ibm.com®**.

Version 1.2
28 November 2018

Contents

List of Figures

List of Tables

Revision Log

Each release of this document supersedes all previously released versions. The revision log lists all significant changes made to the document since its initial release. In the rest of the document, change bars in the margin indicate that the adjacent text was modified from the previous release of this document.

Revision Date	Description
28 November 2018	Version 1.2. • Revised *Appendix A.7 Sampled Instruction Event Register (SIER)* on page 185.
24 August 2018	Version 1.1. • Revised *Figure 5-2 POWER9 Raw Event Coding* on page 125.
17 November 2017	Version 1.05. • Revised *Table 6-2 Event and Performance Metrics for Fabric Events* on page 154. • Revised *Table 6-4 Memory Controller PMU Events and Performance Metrics (Group1)* on page 155. • Revised *Table 6-5 MBA PMU Events and Performance Metrics (Group0)* on page 156. • Revised *Table 6-6 MBA PMU Events and Performance Metrics (Group1)* on page 156. • Removed table previously known as *Table 6-7 CAPP PMU Events and Performance Metrics (Group0)*. • Revised *Table 6-7 XLink PMU Events and Performance Metrics (Group0)* on page 157. • Revised *Table 6-10 Nest Default IMC Groupings* on page 159. • Removed section previously known as *Section 6.4 Performance Co-Pilot for Counter Access*.
31 October 2017	Version 1.0 (initial version).

About this Document

Performance instrumentation is divided into two broad categories: the performance monitor and the trace facilities. The IBM® POWER9™ chip has built-in features for monitoring and collecting data for performance analysis. Collectively, the features are referred to as instrumentation. This document provides a user's view of the POWER9 hardware performance monitoring capability.

Who Should Read This Manual

This manual is intended for system software and hardware developers, application programmers, and analysts who optimize code, tune systems, and characterize workloads.

Document Organization

This document describes the details of the performance monitor features in two basic sections: the core and the nest. *Section 1 - 5* describe the POWER9 core performance monitoring features and *Section 6* describes the POWER9 nest performance monitoring features.

Revision Log	Lists all significant changes made to the document since its initial release.
About this Document	Describes this document, related documents, the intended audience, and other general information.
Core Performance Monitoring Facilities	This section provides a brief overview of the essential core performance monitor functions and facilities.
POWER9 Sampling Support	This section describes the three sampling modes: continuous sampling, random instruction sampling (RIS), and random event sampling (RES).
Thresholding	This sections describes thresholding, which can be used to identify marked instructions that take more than the expected cycles between a start event and an end event.
POWER9 Core	This section provides a brief summary of the POWER9 microarchitecture.
Core PMU Events	This section provides a list of all the PMU events supported on the POWER9 chip for each category and briefly describes which performance bottlenecks are characterized by the events.
Nest PMU Instrumentation and Performance Metrics	This section provides a brief introduction to the Nest units and the events, performance metrics, and formulas for deriving performance metrics from those events.
Performance Monitor Registers	This section describes the performance-monitor related registers.

Conventions

This section explains the number, bit field, instruction, and signal conventions that are used in this document.

Representation of Numbers

Numbers are generally shown in decimal format, unless designated as follows:

- Hexadecimal values are preceded by a 0x.
  For example: 0x600F4.
- Binary values in sentences are shown in single quotation marks.
  For example: '1010'.
- A bit value that is immaterial, which is called a "don't care" bit, is represented by an "x."

Bit Significance

The bit on the left represents the most-significant bit of a field. The bit on the right represents the least-significant bit of a field. For example, in CTL[0:31], 0 is the most-significant bit.

Other Conventions

The following typographical conventions are used in this document.

Convention	Description
lwsync	Instruction mnemonics are shown in lowercase, bold text.
Hyperlink	Web-based URLs are displayed in blue text to denote a virtual link to an external document. For example: http://www.ibm.com
Note: This is note text.	The note text denotes information that emphasizes a concept or provides critical information.
Footnote reference.[1] 1. Descriptive footnote text.	A footnote is an explanatory note or reference inserted at the foot of the page or under a table that explains or expands upon a point within the text or indicates the source of a citation or peripheral information.
Underline	An underline indicates that the definition of an acronym is displayed when the user hovers the cursor over the term.

Related Documents

The documents available in the IBM Portal for OpenPOWER, an online IBM technical library, are helpful in understanding the IBM POWER9 processor. Additional technical resources are available on the OpenPOWER Foundation web site. The following documents are also useful:

- *I^2C Bus Specification (Version 2.1)*
- *PCI Local Bus Specification (Revision 4.0)*

1. Core Performance Monitoring Facilities

The POWER9 chip has built-in features for monitoring and collecting data for performance analysis. Collectively, the features are referred to as instrumentation. Performance instrumentation is divided into two broad categories: the performance monitor and the trace facilities. This section provides a brief overview of the essential core performance monitor functions and facilities.

1.1 Essential Performance Monitor Functions

The POWER9 performance monitor performs the following functions:

- Counts instructions completed and cycles gated by the run latch in individual (dedicated) 32-bit counters. The counting of these events can be enabled by software based on several conditions such as problem or supervisor state and tags active or inactive.
- Counts up to four concurrent software-selected events in individual 32-bit counters.
  The counting of events can be enabled by software based on several conditions such as problem or supervisor state, tags active or inactive mode, and run or wait state.

  One event per counter can be selected for monitoring at a given time. The event to be monitored is selected by setting the appropriate value into the Monitor Mode Control Register (MMCR) event selection fields for that counter. The event counted can be the number of cycles that the event occurs or the number of occurrences of the event depending on the particular event selected. Performance monitor counting can be enabled or disabled under the machine states mentioned previously, which are selected using control bit fields in the MMCRs and the state bits in other Special Purpose Registers (SPRs).
- Generates performance monitor exceptions, alerts, and interrupts.
  The performance monitor can generate a maskable interrupt when an event counter overflows. Additionally, trigger events can cause performance monitor exceptions to occur based on the values of the exception enable bits in the MMCRs. An enabled exception causes an indicator bit to be set in the MMCRs. This bit can only be reset by software. When running in a partitioned environment, the operating system can be swapped out while a performance monitor exception alert is pending. The hypervisor preserves the value of MMCRs across the partition swap. When the operating system is redispatched, the alert is still pending. When enabled for external interrupts, a performance monitor alert causes a performance monitor interrupt to occur.
- Freezes the contents of the event counters until a selected event or condition occurs and then begins counting (triggering).
  The event counters can also be incremented until a selected event or condition occurs, and then counting is frozen.

  Some conditions and events, called trigger events, can be used to control performance monitor activities such as starting and stopping the counters and causing performance monitor exceptions. These scenarios are selected using the condition/event enable bit fields and the exception enable bits of the MMCRs in conjunction with control bits in other SPRs.
- Chooses an instruction for detailed monitoring, which is called sampling or marking an instruction.
  The POWER9 instrumentation supports setting mask values for matching particular instructions or types of instructions, which are then eligible to be sampled. The performance monitor includes events for counting sampled instructions at each stage of the pipeline and for some other situations. Instruction sampling is a useful facility for gathering both detailed and statistical information for particular instructions. Profiling and sampling are common approaches to associate expensive performance events in a processor to instruction and data addresses. Profiling enables the identification of hotspots in code and data, finds per-

formance-sensitive areas, and identifies problem instructions, data areas, or both. Profiling is commonly achieved by identifying a particular instruction and collecting detailed information about that instruction (instruction sampling).

- Performs thresholding. The POWER9 processor monitors the pipeline stage progression of sampled instructions and can detect when the stage-to-stage cycle count for a selected start/stop pair of pipeline stages exceeds a specified threshold value. The threshold value can also be used to detect sampled loads whose latency exceeds the threshold value.

1.2 Definitions and Terminology

Branch history rolling buffer (BHRB)	A buffer that contains a history of branch addresses that have been taken.
Continuous sampling	Sampling every instruction executed and continuously collecting instruction and data addresses.
Eligible instruction	Instructions that are eligible for random sampling.
Event-based sampling	Sampling an instruction based on the occurrence of an event, such as a cache miss or branch mispredict. Randomly mark an instruction after an event has happened (as compared to marking an instruction and hoping that an event happens to that instruction). This feature uses the Sampled Branch Target Address Register in the IFU.
Instruction completion table (ICT)	A table tracking in-flight instructions. Long delays in instruction fetching can cause the ICT to be empty for the currently tracked thread.
Lightweight profiling	User-level interrupts and reduced latency of PMU interrupts.
Marked events	Events attributed to a marked instruction. By convention, marked event names are often annotated with the Mark bit in the VHDL.
Marked instruction	An instruction that has been randomly picked for detailed data collection. This instruction is sometimes referred to as a sampled instruction.
Matched instruction	An instruction that matches the performance entry in the Instruction Match CAM (IMC). Sometimes referred to as IMC marked.
Next-to-complete (NTC)	The program order next-to-complete instruction.
Performance monitor unit (PMU)	The PMU is a programmable component of each microprocessor core on the chip. It collects and filters information collected from various aspects of the chip and attributes the events to the threads within the core.
Random instruction sampling	Randomly picking one instruction on which to collect detailed performance data, including instruction and data addresses. This is also referred to as marking.
Run latch	A mechanism used by hypervisors and operating systems to flag cycles when the processor for a hardware thread is not idle. Run cycles are the cycles the thread is not idle. Run instructions are the instructions executed by the thread when it is not idle.

Sampling	Collecting performance data from a single instruction.
Simultaneous multithreading (SMT)	SMT enables a number of hardware threads to run concurrently on the core. When multiple threads are running concurrently on the core, resources on the core are shared by the threads on the core.
Software-driven marking	ProbeNop is inserted into an instruction stream to mark the following instruction.

1.3 Essential Performance Monitor Facilities

The POWER9 processor instrumentation facilities and the associated POWER9 components include several SPRs associated with performance monitoring, instruction matching, instruction sampling, and tracing. Unless otherwise noted, the special purpose registers described can be read in problem and supervisor state and written in supervisor state by using the **mfspr** and **mtspr** instructions, respectively. The Machine State Register (MSR) is read and written by the **mfmsr** and **mtmsr** instructions.

1.3.1 Performance Monitor Special Purpose Registers and Fields

A high-level overview of the performance-monitor-related registers and fields is provided in this section. Additional details can be found in *Appendix A Performance Monitor Registers* on page 165.

- Performance Monitor Counter Registers (**PMCx**). These registers increment each time (or cycle, depending on the selected event) an event occurs while the counter is enabled. These registers also have the control function for the counter overflow condition. Each thread has six Performance Monitor Counters (PMCs). With four threads per core, each POWER9 core has 24 thread-level Performance Monitor Counters (PMCs). Each PMC is 32 bits wide. By connecting adjacent PMCs, PMC1 - 4 can also be used as a $32 \times N$ (N = 1 - 4) bit counter.
  - PMC1 - 4 are programmable.
  - PMC5 is a dedicated counter for run instructions. Run instructions are completed PowerPC instructions gated by the run latch.
  - PMC6 is a dedicated counter for run cycles. Run cycles are gated by the run latch.
- Performance Monitor Mode Control Registers (**MMCRx**). The performance monitor is configured and controlled through the Monitor Mode Control Registers (MMCRs). These registers include both counting control and event-select bit fields.
  - MMCR0. This partition resource controls basic operation (start/stop/freeze) of the performance monitor.
  - MMCR1. This partition resource controls what to count.
  - MMCR2. This partition resource controls the basic operation of each PMC individually.
  - MMCRA. This partition resource includes indicator bits for feedback between the hardware and software and configuration fields for special features of the performance monitor.
- Sample Address Registers (**SxAR**). These registers can only be updated when performance monitor exceptions are enabled. This protects the contents from change until software can read them. The values written to these registers by the hardware depend on the processing state and on the kind of instruction that is being sampled.

  - Sample Instruction Event Register (SIER). This 64-bit register stores information relating to a sampled or marked instruction.
  - Sampled Instruction Address Register (SIAR). This 64-bit register contains the instruction address relating to a sampled or marked instruction.
  - Sampled Data Address Register (SDAR). This 64-bit register contains the data address relating to a sampled or marked instruction.
- Machine State Register fields related to performance monitoring follow:
  - MSR[EE]. This register bit is used to enable or disable the external interrupt. The performance monitor interrupt is considered an external interrupt.
  - MSR[PMM]. This register bit is used to enable or disable performance monitor activity controlled by the process mark bit.
  - MSR[PR]. This register bit is used to establish problem or supervisor mode and the performance monitor counting activity controlled by this bit.
  - MSR[SE]. This register bit is used to enable or disable a trace interrupt after each instruction is completed.
  - MSR[BE]. This register bit is used to enable or disable a trace interrupt after a branch instruction is completed.
- Control Register[31] (CNTL[31]). This register bit is used by operating systems to indicate an idle or run state. The performance monitor can use this bit to avoid counting events during idle periods. This bit is commonly called the run latch.
- Instruction Match CAM Register (IMC). The IMC SPR is used to access the IMC array, which contains tag bits and mask values used for instruction matching. The **mt/fimc** instructions can be executed only in supervisor mode.
- Timebase[47,51,55,63]. These register bits are used for time-based events. Most performance monitor events are cycle based; that is, they count based on processor cycles. The Timebase register is used to maintain time-of-day and can be used by the performance monitor to count time intervals.
- Machine Status Save/Restore Register (SRRO, SRR1). These registers are used to save the machine status during interrupts.

Table 1-1 on page 16 describes the SPR address bits and the widths for these registers. The subsequent sections describe the various performance-monitor-related registers.

Table 1-1. Performance-Monitor-Related Special Purpose Registers (Sheet 1 of 2)

Register Name	SPR Address Bits[1]			Function
	Decimal (U,P)	[5 - 9] [0^2 - 4]	Width (Bits)	
MMCR0	779,795	'11000' 'n1011'	32	Performance Monitor Mode Control Register 0
MMCR1	782,798	'11000' 'n1110'	64	Performance Monitor Mode Control Register 1
MMCR2	769,785	'11000 'n0001'	64	Performance Monitor Mode Control Register 2
MMCRA	770,786	'11000' 'n0010'	64	Performance Monitor Mode Control Register A

1. In an **mt/fspr** instruction, the instruction SPR field of bits [11:15] hold SPR address bits 0:4 and bits [16:20] hold SPR field bits [5:9].
2. If n = 0, use the user mode **mfspr** instruction SPR address bits. If n = 1, use the privileged mode **mtspr** instruction SPR address value. For **mfspr**, the instruction is in privileged mode if and only if SPR[0] = 1.

Table 1-1. Performance-Monitor-Related Special Purpose Registers (Sheet 2 of 2)

Register Name	SPR Address Bits[1]			Function
	Decimal (U,P)	[5 - 9] [0^2 - 4]	Width (Bits)	
PMC1	771,787	'11000' 'n0011'	32	Performance Monitor Counter Register 1
PMC2	772,788	'11000' 'n0100'	32	Performance Monitor Counter Register 2
PMC3	773,789	'11000' 'n0101'	32	Performance Monitor Counter Register 3
PMC4	774,790	'11000' 'n0110'	32	Performance Monitor Counter Register 4
PMC5	775,791	'11000' 'n0111'	32	Performance Monitor Counter Register 5
PMC6	776,792	'11000' 'n1000'	32	Performance Monitor Counter Register 6
SIER	768,784	'11000' 'n0000'	64	Sampled Instruction Event Register
SIAR	780,796	'11000' 'n01100'	64	Sampled Instruction Address Register
SDAR	781,797	'11000' 'n1101'	64	Sampled Instruction Address Register
MSR[61]	Use **mtmsr**, **mfmsr** instructions (privileged mode only)		64	Machine State Register [Performance Monitor Mark]
MSR[48]			64	Machine State Register [External Interrupt]
MSR[49]			64	Machine State Register [Problem/Supervisor State]
MSR[1]			64	Machine State Register [Tags Active]
MSR[53]			64	Machine State Register [Single-Step Trace Enable]
MSR[54]			64	Machine State Register [Branch Trace Enable]
CTRL[63]	136,152	'00100 'n1000	64	Control Register [Run Latch, Thread Control]
IMC	Use **mtimc**, **mfimc** instructions (privileged mode write, user and privilege mode read)		64	Instruction Match CAM Register
TBL	284	'01000 'n1100	64	Timebase bits used for performance monitor timebase events

1. In an **mt/fspr** instruction, the instruction SPR field of bits [11:15] hold SPR address bits 0:4 and bits [16:20] hold SPR field bits [5:9].
2. If n = 0, use the user mode **mfspr** instruction SPR address bits. If n = 1, use the privileged mode **mtspr** instruction SPR address value. For **mfspr**, the instruction is in privileged mode if and only if SPR[0] = 1.

Table 1-2 describes the performance-monitor-counter properties.

Table 1-2. Performance Monitor Counter (PMC) Properties

PMC	Programmable by Partition	Programmable by Hypervisor	Partition Access	Hypervisor Access	Exception on Overflow	Interrupt Destination	Width (Bits)
PMC1	Yes	Yes	Read/Write	Read/Write	Yes	Partition	32
PMC2	Yes	Yes	Read/Write	Read/Write	Yes	Partition	32
PMC3	Yes	Yes	Read/Write	Read/Write	Yes	Partition	32
PMC4	Yes	Yes	Read/Write	Read/Write	Yes	Partition	32
PMC5	No (finished instruction count gated by run latch only)		Read/Write	Read/Write	Yes	Partition	32
PMC6	No (cycle count gated by run latch only)		Read/Write	Read/Write	Yes	Partition	32

2. POWER9 Sampling Support

Profiling or sampling is a common approach to associate expensive performance events in a processor to instruction and data addresses. Profiling enables the identification of hotspots in code and data, finds performance-sensitive areas, and identifies problem instructions, data areas, or both. Profiling is commonly achieved by identifying a particular instruction and collecting detailed information about that instruction (instruction sampling).

The Power ISA provides two SPRs to identify sampled instructions: the Sampled Instruction Address Register (SIAR) and the Sampled Data Address Register (SDAR). An indicator bit in the Monitor Mode Control Register A (MMCRA) indicates when the SIAR and SDAR are from the same instruction. The SDAR is not cleared when a new sampled instruction is selected. Therefore, the indicator bit is required to show that the SDAR is not for a previously sampled instruction (that is, for an executed or cancelled instruction). The sampled registers can only be updated by the processor when performance monitor exceptions are enabled. Performance monitor exceptions toggle the enable bit, locking the contents of the sampled registers. This makes it possible to profile code not enabled for interrupts.

The POWER9 processor supports the following three sampling modes:

- Continuous sampling collects information on every instruction completion and data-cache reload. Continuous sampling is useful for profiling on execution frequency or cache-line accesses.
- Random instruction sampling (RIS) selects (or marks) one instruction at a time and tracks its execution through the processor pipeline. Events that can be attributed to a sampled instruction are called marked events. By profiling on marked events, it is possible to uniquely identify which instruction caused a particular event.
- Random event sampling (RES) selects or marks an instruction after an event has happened to an instruction. This mode improves sampling rates for some very important performance-sensitive events, such as branch mispredicts.

While sampling is extremely useful, processors since the IBM POWER6 design have had the ability to associate multiple performance events to the same instruction or data address. This is accomplished by recording up to 64 bits of information pertaining to a marked instruction during its lifetime in the pipeline. These 64 bits are accessible via a software-accessible Sampled Instruction Event Register (SIER), which enables the collection of multiple events with a single pass.

2.1 Sampled Instruction Address Register

The SIAR is a 64-bit register that contains the effective address of the sampled instruction. When continuous sampling is enabled by MMCRA[SAMPLE_ENABLE], all instructions are sampled. When a performance monitor alert occurs, the SIAR contains the effective address of the instruction that was being executed, possibly out of order, at or around the time that the performance monitor alert occurred.

When random sampling is enabled by MMCRA[SAMPLE_ENABLE], only those instructions specified by MMCRA list fields are sampled. When a performance monitor alert occurs, the SIAR contains the effective address of the last sampled instruction that had completed when the performance monitor alert occurred. The contents of SIAR can be altered by the hardware, if and only if, MMCR0[PMAE] = '1'. Therefore, after the performance monitor alert occurs, the contents of SIAR are not altered by the hardware until software sets MMCR0[PMAE] to '1'. After software sets MMCR0[PMAE] to '1', the contents of SIAR are undefined until the next performance monitor alert occurs.

Programming Note: If the performance monitor alert causes a performance monitor interrupt, the value of MSR[HV, PR] that was in effect when the sampled instruction was being executed is reported in the SIER. Since the POWER8 processor (DD2.0 and beyond), the value of MSR[HV, PR] is also be reflected in the SIER in continuous-sampling mode.

Engineering Note: If the performance monitor alert is caused by an enabled counter-negative condition that can be associated with the execution of a specific instruction, it is preferable to set SIAR to that instruction's address.

2.2 Sampled Data Address Register

The SDAR is a 64-bit register that contains the effective address of the sampled data when a performance monitor alert occurs. When a performance monitor alert occurs, the SDAR is set to the effective address of the storage operand of an instruction that was being executed, possibly out-of-order, at or around the time that the performance monitor alert occurred. This storage operand is called the sampled data. The sampled data can be, but is not required to be, the storage operand (if any) of the sampled instruction.

2.3 Continuous Sampling

The POWER9 processor supports continuous sampling where the SIAR and SDAR are continuously loaded, as long as PMAE = '1', MMCRA[63] = '0', and MSR[BE, SE] = '00'.

The SIAR is loaded at completion time with the effective address of the youngest instruction.

The SDAR depends on the SDAR_MODE bits in the MMCRA Register. See *Appendix A.6 Monitor Mode Control Register A (MMCRA)* on page 181 for the bit descriptions.

SIER[SIAR_VALID] and SIER[SDAR_VALID] are tied to '0' when in continuous sampling mode.

2.4 Random Instruction Sampling

The POWER9 processor supports instruction-based sampling, where an instruction is randomly picked during group formation based on eligibility criteria, and the SIAR is loaded with the exact instruction address at dispatch time for that marked instruction. If applicable, the SDAR is loaded with the data effective address by the load store unit.

As shown in *Table 2-1*, sampling is enabled when MMCRA[63] = '1'. Random instruction sampling is selected when MMCRA[61:62] = '00'. Additional sampling criteria are selected in MMCRA[57:59].

Table 2-1. Random Instruction Sampling Modes

MMCRA[63] SAMPLE_ENABLE	MMCRA[61:62] RAND_SAMP_MODE	MMCRA[57:59] RAND_SAMP_ELIG	Eligibility Criteria
1	00	000	All instructions are eligible for sampling.
1	00	001	All load/store iops are eligible.
1	00	010	All ProbeNops are eligible.
1	00	011	Reserved.
1	00	100	IMC.
1	00	101	IMC and random sampling.
1	00	110	Long latency operation (**div**/**sqrt**/**mul**/**mtctr**/**br** LK = 1).

A marked instruction can generate a number of marked events that are configurable on PMCs 1 - 4. This can be achieved as follows:

- Configure MMCRA to pick random instruction sampling with the eligibility criteria specified.
- Configure a marked event in one of the PMCs and load a threshold (2^{31} - threshold) in the PMC.
- Upon counter overflow, an interrupt occurs. The SIAR, SDAR, and SIER contain information about the marked instruction, which caused an event that caused the counter to overflow.

2.4.1 Value Profiling Support

Value profiling is a key technique used by static and runtime compilers for optimizations such as:

- Determining loop bounds for loop optimizations
- Speculation on values for loads that always return the same value
- Code specialization for particular values
- Memory disambiguation

Current software techniques for value profiling require expensive software instrumentation and can lead to significant overhead for collecting data that might negate the effect from the optimization performed.

The POWER9 design offers the following features to enable value profiling with minimal overhead.

2.4.1.1 ProbeNop

The architecture provides two special nops called ProbeNop defined as ***and 0,0,0*** and ***and 1,1,1***. This form of ***and*** is reserved exclusively for performance monitor use. Software can insert ProbeNop instructions at various points in the program and configure the performance monitor sampling facility to mark or sample only ProbeNops. The sampled ProbeNops can then be configured to count in a PMC and, upon overflow, cause a synchronous PMU interrupt. This enables the capability to freeze the state of the thread after the completion of a ProbeNop so that interrupt handlers can examine general purpose registers, SPRs, memory locations to read stack spills, global variables, and so on.

An example of specifying a ProbeNop follows. The instrumentation point can sample the loop upper-bound values (NX,NY) and a value VAL.

```
for (i=0;i < NX;i++) {
    for(j =0;j < NY;j++) {
        <expression VAL>
    probeNop; // to read NX,NY,VAL
    }
}
```

2.4.1.2 Synchronous PMU Interrupts

PMU interrupts typically occur because of counter overflows. Counter overflows happen because an event occurred in the processor and the counter's bits are all '1' (cannot increment anymore). In legacy processor designs, the PMU interrupts have been asynchronous, meaning the interrupt was delivered many cycles after an event occurred on the processor.

In the POWER9 processor, one of the focus items for software performance is value profiling. The POWER9 PMU has significant support for value profiling, by supporting synchronous interrupts for some selected events.

The goal of a synchronous interrupt is to take an interrupt after an instruction that caused an event, which caused a counter overflow to occur, has completed. This mechanism preserves the state of the thread around the instruction so that all register values are preserved for interrupt handlers to examine.

Synchronous interrupts are only supported for instructions marked using random instruction sampling (RIS). These events are available only on PMC1 and cannot be configured to count simultaneously. Software can configure any of these events and pre-load the PMC with a count and expect to get a synchronous PMU interrupt. The hardware also sets status bit MMCR0[PMAQ] = '1' to indicate that the interrupt was synchronous. To support synchronous interrupts, the hardware must alter instruction flow; therefore, some overhead is involved. However, this overhead is small compared to the software instrumentation required to gather profile data. Specifically, when synchronous interrupts are enabled and the PMU is primed for a synchronous interrupt (that is, close to an overflow), instruction decode prevents forming groups with instructions beyond a taken branch. The completion unit also prevents marked instructions from completing until appropriate handshakes are performed.

2.5 Random Event Sampling

Random instruction sampling is very effective at providing coverage of a wide variety of interesting events. However, it might not be able to provide high enough sample rates for some very delinquent events.

For example, if we are interested in branches that mispredict, RIS can be set up with eligibility criteria to mark only branches. Typically, 95 - 98% of all branches are predicted correctly. Therefore, the chances of sampling a branch that mispredicts is low. To support dynamic optimizers, which require much higher sampling rates to make optimization decisions, there exists a requirement to randomly mark only branch mispredicts (filter out all the hits). The SIAR is updated at completion time with the exact instruction address of the marked instruction, and the SDAR is updated with the data effective address of marked instruction.

2.5.1 Random Event Sampling (RES) in a Branch Unit

Table 2-2 shows the random event sampling modes in the branch unit.

Table 2-2. Random Event Sampling Modes in the Branch Unit

MMCRA[63] SAMP_ENABLE	MMCRA[61:62] RAND_SAMP_MODE	MMCRA[57:59] RAND_SAMP_ELIG	Random Event Sampling Mode Description
1	10	000	Branch mispredicts.
1	10	001	Branch mispredicts (CR).
1	10	010	Branch mispredict (TA).
1	10	011	Taken branches.
1	10	100	Nonrepeating branches
1	10	101	All branches that require prediction.

2.5.2 Random Instruction and Event Selection

As shown in *Table 2-1* on page 21 and *Table 2-2* on page 23, sampling is enabled when MMCRA[63] = '1'. Random instruction sampling is selected when MMCRA[61:62] = '00', and random event sampling is selected when MMCRA[61:62] = '10'. Additional sampling criteria are selected in MMCRA[57:59].

The MMCR1 unit and PMCxSEL are configured for a PMC counter to count stall cycles for a particular marked event. That counter's bits 1:31 are initialized to the 1's complement threshold number of stall cycles required to cause an interrupt. When the interrupt occurs, a program can read interesting data about the marked instruction captured in the SIAR, SDAR, and SIER registers.

3. Thresholding

Profiling is a performance analysis technique that associates hardware performance events with code. It provides a mechanism for application developers to precisely identify in code where cache misses, translation misses, branch mispredicts, and other expensive hardware events occur, so that they can take corrective action. The POWER9 PMU provides support to mark or sample instructions randomly and locks in the instruction effective address and data effective address of those marked events, which enables profiling. The POWER9 PMU also supports the ability to count the number of events generated between a designated start/end marked event in the pipeline and the memory subsystem. For example, application developers can profile on marked/sampled cache misses and identify code where cache misses are most frequent. While this is useful, some situations require the ability to profile on cache misses that take more than the expected number of cycles to resolve.

Thresholding, in this case, can be used to identify marked instructions that take more than the expected cycles between a start event (MARKED CACHE MISS) and an end event (MARKED CACHE RELOAD). Software can specify a threshold and only look at cache misses that exceed a threshold. The same technique can be used to threshold on pipeline stages, such as a decode, dispatch, issue, finish, completion, and also for stores until the memory subsystem has completed the store.

The POWER9 PMU has expanded on this capability with the following features:

1. Ability to count the number of events between a start event and an end event.
2. Ability to specify a threshold to compare against the count of events between a start event and an end event.
3. Ability to specify any event programmable in PMC1 - PMC4, apart from cycles to count between a start event and an end event.

Hardware supports the following capabilities:

1. 10-bit software-accessible floating-point counter with mantissa and exponent
2. 10-bit software-accessible floating-point threshold with mantissa and exponent
3. 4-bit software-accessible threshold start event
4. 4-bit software-accessible threshold end event
5. Software-accessible bits (3 bits) to specify events to threshold on
6. 10-bit hidden auto prescalar

3.1 Floating-Point Counter and Threshold Operation

The POWER9 PMU implements a floating-point counter to count the number of events elapsed between a marked/sampled start and end event. Hardware provides a 7-bit mantissa and a 3-bit exponent for both a counter and a compare facility.

The floating-point counter begins counting from zero, starting after an event specified by a threshold event start and ending at the event specified by threshold event end in the MMCRA. At the start, the auto-prescalar is set at divide by 1 of the event specified by the event selector. Each time the 7-bit counter mantissa overflows, the exponent is incremented by 1, and the counter mantissa continues counting from the new value '0100000'. This ensures that a subsequent read of the counter (exponent and mantissa) is a "base 4" floating-point number. Each time the exponent changes, a new pre-scale divide factor is set in the auto prescalar according to *Table 3-1* on page 26.

To get a fixed-point value from the counter, read the counter mantissa and shift it left two bits, the number of times specified by the counter exponent.

To write the threshold in MMCRA, take a 17-bit number (N), shift it right two places, and increment the exponent (E) by 1 until the upper 10 bits of N are zero. Write E to the threshold exponent, and write the lower 7 bits of N to the threshold mantissa. Note that it is invalid to write a mantissa with the upper two bits of mantissa being zero, unless the exponent is also zero. The maximum threshold that can be written is 130,048.

Table 3-1. Floating-Point Counter Values

Exponent	Prescalar Divide By	Minimum Count Value	Maximum Count Value
0	1	0	127
1	4	128	508
2	16	512	2032
3	64	2048	7936
4	256	8192	32512
5	1024	32768	130048

3.1.1 Thresholding Operation

Software programs the MMCRA to specify the following:

1. Enable sampling, with any eligibility required, using MMCRA[57:63]. These bits enable the sampling state machine, which is a pre-requisite for thresholding.
2. Enable a threshold start event (MMCRA[48:51]) and threshold stop event (MMCRA[52:55]). When MMCRA[63] = '0', no thresholding event can happen.
3. If required, software can also specify a threshold value to compare against, as specified in the MMCRA.
4. Enable thresholding by specifying an event to threshold in (MMCRA[45:47]). The event can be cycles during which the run latch is set, instructions completed while the run latch is set, or any event programmable in PMC1 - PMC4 or as specified in MMCRA (dependent on all freeze conditions including the run latch).

When a threshold start event is encountered, the logic starts incrementing the threshold counter from zero. The counter stops counting if either a threshold stop (and not a start) condition occurs, the sampled instruction is flushed, or if the counter hits the maximum value. Subsequent start conditions reset the threshold counter, and threshold event counting resumes. If the threshold value specified in the compare field of the MMCRA register matches the counter, the SIER bit for threshold exceeded is set. Software can also specify an event called THRESHOLD EXCEEDED to count the number of occurrences.

For example, if software intends to capture the latency of a cache miss, it can select the start event to be MARKED CACHE MISS and the stop event to be MARKED CACHE RELOAD. Software can set up a PMC to count MARKED CACHE RELOADs, and the interrupt handler can capture the instruction and data effective address from SIAR/SDAR and look at MMCRA to capture the cycles between miss and reload.

The software can also select events called THRESHOLD_EXCEEDED and THRESHOLD_NOT_EXCEEDED to profile on for the cases of interest.

A threshold start event takes precedence over a threshold end event. For example, if the threshold state machine is idle and both a start and end event occur on the same cycle, the threshold state machine ignores the end event while recognizing the start event. The threshold state machine transitions from idle to running.

If running, and both a start event and an end event occur on the same cycle, the start takes precedence so that the threshold state machine stays running and the threshold counter is reset to zero. Note that, whenever a threshold start event occurs, the threshold counter is reset to zero, regardless of whether an end event occurs on the same cycle. Because a threshold start is restarting another threshold event counting period, threshold met and threshold exceeded events do not occur when a threshold start is active.

All of the start/end threshold events are provided directly to the performance monitor, which means that the thresholding facility is available regardless of the configuration of the event bus. The events that might be used for threshold start/end measurement occur for a marked instruction moving through the pipeline in the following order.

Events that are available for threshold start and stop are categorized as follows:

For all instructions:

1. Marked instruction decoded
2. Marked instruction dispatched
3. Marked instruction Issued
4. Marked instruction finished
5. Marked group next-to-complete
6. Marked instruction next-to-finish
7. Marked instruction complete

For Loads:

1. Marked instruction decoded
2. Marked instruction dispatched
3. Marked instruction issued
4. Marked load miss
5. L2 RC machine dispatched for marked instruction
6. L2 RC machine done for marked instruction
7. Marked load reload
8. Marked instruction finished
9. Marked group next-to-complete
10. Marked instruction next-to-finish
11. Marked instruction complete

For Stores:

1. Marked instruction decoded
2. Marked instruction dispatched
3. Marked instruction Issued
4. Marked instruction finished
5. Marked group next-to-complete
6. Marked instruction next-to-finish
7. Marked instruction complete
8. L2 RC machine dispatched for marked instruction
9. L2 RC machine done for marked instruction

The PMU also provides the following events to be used along with the thresholding facility:

- PM_THRESH_MET
  This event is active in the cycle subsequent to the threshold counter becoming equal to or greater that the threshold compare value (MMCRA[THRESH_CMP_EXP] and MMCRA[THRESH_CMP_MANTISSA]).
- PM_THRESH_NOT_MET
  A threshold end event has occurred, and the threshold compare value has not been met. This is useful for cases when instructions completed between a load miss and a cache reload. (Lower values are worse than higher values.)
- Threshold Exceeded Events
  The threshold exceeded events are active for one cycle as the threshold counter crosses from less than to equal to or greater than the compare value. These events provide coarse grain thresholding capability on multiple ranges without requiring the specification of a high-precision compare value. Up to four events (PMC1 - PMC4) can be specified. These include the following events:
  - PM_THRESH_CTR_EXC_32
  - PM_THRESH_CTR_EXC_64
  - PM_THRESH_CTR_EXC_128
  - PM_THRESH_CTR_EXC_256
  - PM_THRESH_CTR_EXC_512
  - PM_THRESH_CTR_EXC_1024
  - PM_THRESH_CTR_EXC_2048
  - PM_THRESH_CTR_EXC_4096

3.1.2 Examples of the Ability to Change Events to Threshold

Any event configurable in PMC1 - PMC4 can be used to count between a start and stop event in POWER9 thresholding. A few examples are shown that illustrate how to configure different events with different start and stop pairs to measure different metrics.

3.1.2.1 Loads

In this case, the start event is a marked/sampled cache miss and the end event is a marked cache reload. <

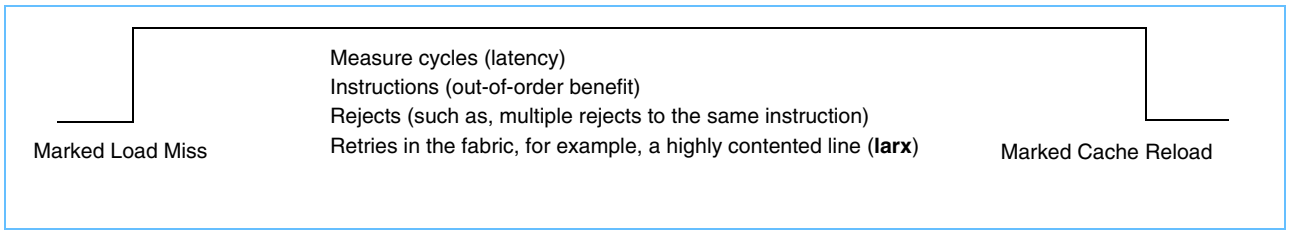
By selecting to count cycles between these two events, the number of cycles elapsed can be counted between a miss and reload. This delta is termed latency. Users can count instructions completed by this thread while this miss was pending, which indicates an out-of-order benefit. If, between the start and stop event, the thread was able to complete many instructions, this is an indication that the pipeline was able to hide the latency of the cache miss. If no instructions were completed, the pipeline was waiting on the cache miss to finish and was unable to hide the latency.

Other useful events to count between a miss and reload are rejects that a marked load incurs. Sometimes a load incurs multiple rejects. This facility can be used to filter out load misses that incurred multiple misses, such as LMQ rejects. The start event can be changed to a marked instruction issued. The stop event can be marked cache reload to count other types of rejects such as issue rejects, ERAT miss, store-hit-load, and so on.

The POWER9 PMU also has the expanded capability to tie fabric-level events to thread-level data. Users can configure software to count fabric retries and threshold on those that take too long. This also appears as added latency.

A marked load miss is the threshold start event and a cache reload is the stop event. *Figure 3-1* shows how to measure cycles between these events. The cycles can be from cache latency, rejects, retries, and so on.

Figure 3-1. Marked Load Events



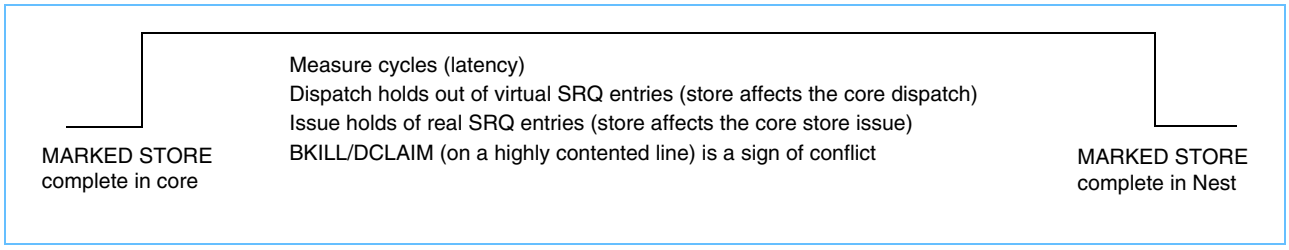
3.1.2.2 Stores

Characterizing long latency stores is difficult. From a core-stall perspective, stores complete and the core can progress towards completing younger instructions. However, if there are many stores that take too long to complete in the L2 cache, this can cause back pressure and the core can run out of virtual and real SRQ entries.

The POWER9 PMU can filter out stores that take too long in the nest. This is important because stores can take hundreds of cycles in the L2 cache but might not impact performance.

A marked store (completed in the core) is the threshold start event and the marked store (completed in the Nest) is the stop event. *Figure 3-2* shows how to measure cycles between these events.

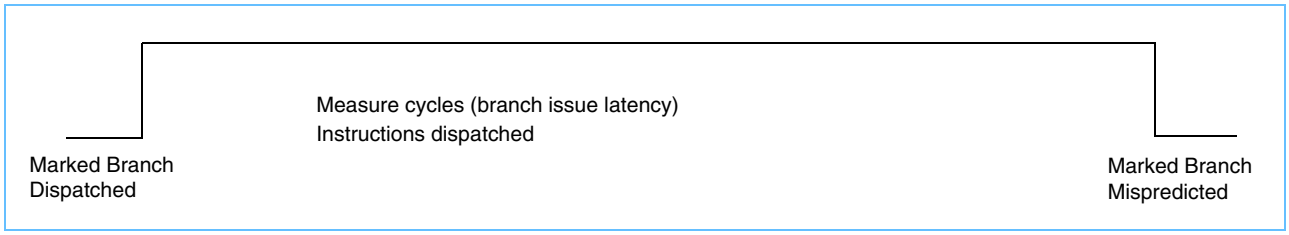
Figure 3-2. Marked Store Events



3.1.2.3 Branches

A marked branch dispatched is the threshold start event and a marked branch mispredicted is the stop event. *Figure 3-3* shows how to measure cycles between these events.

Figure 3-3. Branches



3.1.3 POWER9 Threshold Event Selection

See *Appendix A.6 Monitor Mode Control Register A (MMCRA)* on page 181 for a description of the threshold event selection field (MMCRA[45:47]) when used to select an event specified for threshold counting.

3.1.4 POWER9 Threshold Start/Stop Event Selection

Table 3-2 lists the threshold start and stop event conditions.

Table 3-2. POWER9 Threshold Start/Stop Event Selection

MMCRA[48] MMCRA[52]	MMCRA[49:51] MMCRA[53:55]	Description	Architected
0	000	No start/stop event.	Yes
0	001	Sampled instruction decoded.	Yes
0	010	Sampled instruction dispatched.	Yes
0	011	Sampled instruction issued.	Yes
0	100	Sampled instruction finished.	Yes
0	101	Sampled instruction completed.	Yes
0	110	Sampled instruction L1 load miss.	Yes
0	111	Sampled instruction L1 reload.	Yes
1	000	PMC1 event (depends on all PMC1 freeze conditions).	No
1	001	PMC2 event (depends on all PMC1 freeze conditions).	No
1	010	PMC3 event (depends on all PMC1 freeze conditions).	No
1	011	PMC4 event (depends on all PMC1 freeze conditions).	No
1	100	Sampled group next-to-complete.	No
1	101	RC machine dispatched for sampled instruction.	No
1	110	RC machine done for sampled instruction.	No
1	111	Sampled instruction next-to-finish.	No

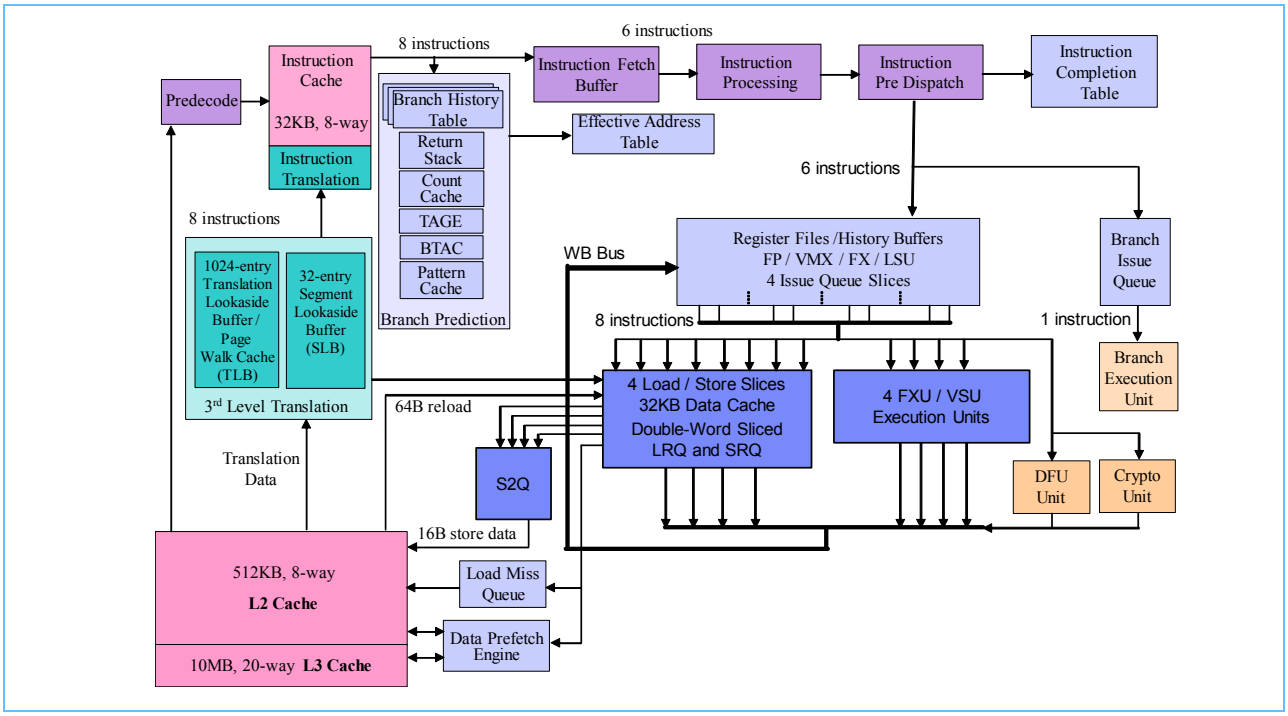
4. POWER9 Core

The POWER9 core supports in excess of 1000 events that help to collect and filter information from across the core. These events provide insight into how the instructions in the program flow in the POWER9 pipeline and help understand any performance bottlenecks seen in the core. This section provides a brief summary of the POWER9 microarchitecture. *Section 5 Core PMU Events* on page 34 provides a summary of the events and rules for grouping events together for measurement.

4.1 POWER9 Core Features

The POWER9 processor is an out-of-order issue processor with a speculative superscalar inner core design supporting 4-way simultaneous multi-threading. The POWER9 core is a 4-slice microarchitecture and is significantly different from the POWER8 core. *Figure 4-1* shows a block diagram of the POWER9 core.

Figure 4-1. POWER9 Processor Core



Some of the key features include:

- Multi-threaded core design
  - Single thread (ST), 2-way multi-thread (SMT2), and 4-way multi-thread (SMT4).
  - Four logical partitions (LPARs) supported at a time.
- Aggressive branch prediction
  - Prediction for up to eight branches per cycle.
  - Support for up to 40 predicted taken branches in-flight, ST mode. Twenty predicted taken branches per thread in SMT2 mode and ten predicted taken branches per thread in SMT4 mode.
  - Prediction support for branch direction and branch target addresses.

- In-order dispatch of up to six internal operations (iops) into five distributed issue queues per cycle
  - Up to two branches dispatched per cycle.
  - Up to six non-branch instructions dispatched per cycle.
- Out-of-order issue of up to nine operations
  - Four load or store agen operations.
  - Four 64-bit execution or computational operations; 128-bit operations are issued as a pair of 64-bit issues.
  - One branch operation.
- Register renaming on GPRs, FPRs, CR fields, XER (parts), FPSCR, VSCR, Link, TAR, and Count
- Eleven execution units
  - Four symmetric load/store units (LSU).
  - Four symmetric 64-bit VMX execution units capable of executing fixed point ALU, simple FX, complex FX, permute, 128-bit fixed-point, single-precision, double-precision, floating-point operations. Two execution units are tied together to perform 128-bit execution.
    - Four floating-point units (FPU). Each FPU supports a double-precision operation or up to two single-precision operations each for SIMD and also supports fixed-point multiply and complex FX operations.
    - For each symmetric unit, only one operation per cycle can be issued.
  - One decimal floating-point and quad-precision floating-point unit (DFU).
- One crypto unit.
- One branch execution unit (BR).
- A large number of instructions in flight.
  - 96 instructions deep, instruction-fetch buffer, split equally across the threads in SMT2 and SMT4 mode
  - Up to 24 instructions in four dispatch pipe stages
  - Up to 256 instructions from through instruction completion
  - Up to 64 stores queued in the SRQ (available for forwarding), shared by the available threads and buffered en-route to the L2 cache through a 16 entry S2Q.
- Fast, selective flush of incorrect speculative instructions and results

4.2 Pipeline Structure

The pipeline structure for the microprocessor can be subdivided into a master pipeline and several different execution unit pipelines. The master pipeline presents speculative in-order instructions to the mapping, sequencing, and dispatch functions, and ensures an orderly completion of the real execution path (throwing away any other potential speculative results associated with mispredicted paths). The execution unit pipelines allow out-of-order issuing of both speculative and non-speculative operations. The execution unit pipelines progress independently from the master pipeline and from one another.

Figure 4-2. Pipeline Structure

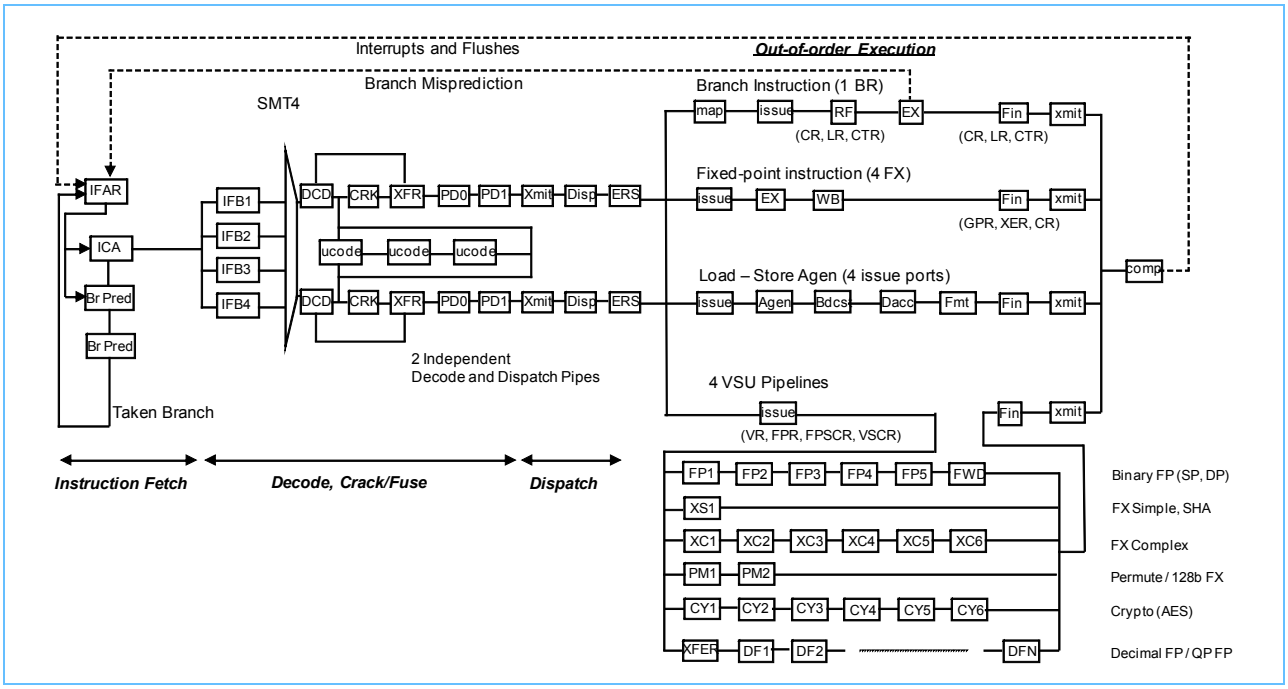


Figure 4-2 illustrates these pipelines, where each box represents a pipeline stage. Definitions for *Figure 4-2* are as follows:

CA/fmt	Data cache access and data formatting	GCT	Global completion table
comp	Group completion	ICA	Instruction cache access
CRK	Crack/Fuse	IFAR	Instruction Fetch Address Register
DCD	Decode	ifbx	Instruction fetch buffer latches
DISP	Dispatch	pdx	Pre-dispatch
DSP	Group dispatch	ucode	Microcode
EA/CA	Effective address generation and data cache decode	WB	Writeback to the register file
ERS	Issue queue	WRT	Format and write into the GCT
EX	Execution	XFR	Transfer
FP1	Floating-point alignment and multiply'	xmit	Finish and transmit
FP2	Multiply	Xmit	Instruction transfer to dispatch
FP3	Add		
FP4	Normalize result		
FP5	Round result and re-drive		

5. Core PMU Events

The POWER9 core supports in excess of 1000 events that collect and filter information from across the core. These events provide insight into how the program instructions flow in the POWER9 pipeline and assist in understanding any performance bottlenecks seen in the core.

The following sections provide a list of all the PMU events supported on the POWER9 chip for each category and briefly describes which performance bottlenecks are characterized by the events. The event tables show the PMC that counts the event, an event code that uniquely identifies an event, the event name, and the event description,. The event code value is used by software tools (for example, PERF) to program the various MMCR registers to read the events on the PMCs.

The events lists are organized as follows:

5.1 IFU Events

Table 5-1 lists the POWER9 events related to the instruction cache and other fetch-related (IFU) events.

Table 5-1. IFU Events (Sheet 1 of 4)

PMC	Event Code	Event Name	Event Description
PMC1	0000010002	PM_INST_CMPL	Number of PowerPC instructions that completed.
PMC1	0000010018	PM_IC_DEMAND_CYC	I-cache miss demand cycles.
PMC1	0000014040	PM_INST_FROM_L2_NO_CONFLICT	The processor's I-cache was reloaded from the local core's L2 cache without conflict due to an instruction fetch (not prefetch).
PMC1	0000014042	PM_INST_FROM_L2	The processor's I-cache was reloaded from the local core's L2 cache due to an instruction fetch (not prefetch).
PMC1	0000014044	PM_INST_FROM_L3_NO_CONFLICT	The processor's I-cache was reloaded from the local core's L3 cache without conflict due to an instruction fetch (not prefetch).
PMC1	0000014046	PM_INST_FROM_L3.1_SHR	The processor's I-cache was reloaded with shared (S) data from another core's L3 cache on the same chip due to an instruction fetch (not prefetch).
PMC1	0000014048	PM_INST_FROM_ON_CHIP_CACHE	The processor's I-cache was reloaded either shared or modified data from another core's L2/L3 on the same chip due to an instruction fetch (not prefetch).
PMC1	000001404A	PM_INST_FROM_RL2L3_SHR	The processor's I-cache was reloaded with shared (S) data from another chip's L2 or L3 cache on the same node or group (remote), as this chip due to an instruction fetch (not prefetch).
PMC1	000001404C	PM_INST_FROM_LL4	The processor's I-cache was reloaded from the local chip's L4 cache due to an instruction fetch (not prefetch).
PMC1	000001404E	PM_INST_FROM_L2MISS	The processor's I-cache was reloaded from a location other than the local core's L2 cache due to an instruction fetch (not prefetch).
PMC1	0000014050	PM_INST_CHIP_PUMP_CPRED	Initial and final pump scope was chip pump (prediction is correct) for an instruction fetch.
PMC1	0000014052	PM_INST_GRP_PUMP_MPRED_RTY	Final pump scope (group) ended up larger than the initial pump scope (chip) for an instruction fetch.
PMC1	0000014054	PM_INST_PUMP_CPRED	Pump prediction correct. Counts across all types of pumps for an instruction fetch.
PMC1	00000100FE	PM_INST_CMPL	Number of PowerPC instructions finished (completed).
PMC2	0000020002	PM_INST_CMPL	Number of PowerPC instructions finished (completed).
PMC2	0000024040	PM_INST_FROM_L2_MEPF	The processor's I-cache was reloaded from the local core's L2 hit without dispatch conflicts on an Mepf state due to an instruction fetch (not prefetch).
PMC2	0000024042	PM_INST_FROM_L3_MEPF	The processor's I-cache was reloaded from the local core's L3 cache without dispatch conflicts hit on an Mepf state due to an instruction fetch (not prefetch).
PMC2	0000024044	PM_INST_FROM_L3.1_MOD	The processor's I-cache was reloaded with modified (M) data from another core's L3 cache on the same chip due to an instruction fetch (not prefetch).
PMC2	0000024046	PM_INST_FROM_RL2L3_MOD	The processor's I-cache was reloaded with modified (M) data from another chip's L2 or L3 cache on the same node or group (remote) as this chip due to an instruction fetch (not prefetch).
PMC2	0000024048	PM_INST_FROM_LMEM	The processor's I-cache was reloaded from the local chip's memory due to an instruction fetch (not prefetch)
PMC2	000002404A	PM_INST_FROM_RL4	The processor's I-cache was reloaded from another chip's L4 cache on the same node or group (remote) due to an instruction fetch (not prefetch).
PMC2	000002404C	PM_INST_FROM_MEMORY	The processor's I-cache was reloaded from a memory location including the L4 cache from a local remote or distant due to an instruction fetch (not prefetch).

Table 5-1. IFU Events (Sheet 2 of 4)

PMC	Event Code	Event Name	Event Description
PMC2	000002C05C	PM_INST_GRP_PUMP_CPRED	Initial and final pump scope was group pump (prediction is correct) for an instruction fetch (demand only).
PMC2	000002C05E	PM_INST_GRP_PUMP_MPRED	Final pump scope (group) ended up either larger or smaller than the initial pump scope for an instruction fetch (demand only).
PMC2	00000200F2	PM_INST_DISP	Number of PowerPC instructions dispatched.
PMC2	00000200FD	PM_L1_ICACHE_MISS	Demand I-cache miss.
PMC3	0000030002	PM_INST_CMPL	Number of PowerPC instructions finished (completed).
PMC3	0000034040	PM_INST_FROM_L2_DISP_CONFLICT_LDHITST	The processor's I-cache was reloaded from the local core's L2 cache with a load-hit-store conflict due to an instruction fetch (not prefetch).
PMC3	0000034042	PM_INST_FROM_L3_DISP_CONFLICT	The processor's I-cache was reloaded from the local core's L3 cache with dispatch conflict due to an instruction fetch (not prefetch).
PMC3	0000034044	PM_INST_FROM_L3.1_ECO_SHR	The processor's I-cache was reloaded with shared (S) data from another core's ECO L3 on the same chip due to an instruction fetch (not prefetch).
PMC3	0000034046	PM_INST_FROM_L2.1_SHR	The processor's I-cache was reloaded with shared (S) data from another core's L2 cache on the same chip due to an instruction fetch (not prefetch).
PMC3	0000034048	PM_INST_FROM_DL2L3_SHR	The processor's I-cache was reloaded with shared (S) data from another chip's L2 or L3 cache on a different node or group (distant) from this chip due to an instruction fetch (not prefetch).
PMC3	000003404A	PM_INST_FROM_RMEM	The processor's I-cache was reloaded from another chip's memory on the same node or group (remote) due to an instruction fetch (not prefetch).
PMC3	000003404C	PM_INST_FROM_DL4	The processor's I-cache was reloaded from another chip's L4 cache on a different node or group (distant) due to an instruction fetch (not prefetch).
PMC3	0000034050	PM_INST_SYS_PUMP_CPRED	Initial and final pump scope was a system pump (prediction is correct) for an instruction fetch.
PMC3	0000034052	PM_INST_SYS_PUMP_MPRED	Final pump scope (system) mispredicted. Either the original scope was too small (chip/group) or the original scope was system and it should have been smaller. Counts as an instruction fetch.
PMC3	0000030068	PM_L1_ICACHE_RELOADED_PREF	Counts all I-cache prefetch reloads (includes a demand turned into a prefetch).
PMC3	00000300F2	PM_INST_DISP	Number of PowerPC instructions dispatched.
PMC3	00000300F4	PM_THRD_CONC_RUN_INST	Number of PowerPC instructions finished by this thread when all threads in the core had the run-latch set.
PMC3	00000300FA	PM_INST_FROM_L3MISS	Marked instruction was reloaded from a location beyond the local chiplet.
PMC4	0000040002	PM_INST_CMPL	Number of PowerPC instructions completed
PMC4	0000040012	PM_L1_ICACHE_RELOADED_ALL	Counts all I-cache reloads; includes demand, prefetch, prefetch turned into demand, and demand turned into prefetch.
PMC4	000004001C	PM_INST_IMC_MATCH_CMPL	IMC match count.
PMC4	0000044040	PM_INST_FROM_L2_DISP_CONFLICT_OTHER	The processor's I-cache was reloaded from local core's L2 cache with a dispatch conflict due to an instruction fetch (not prefetch).
PMC4	0000044042	PM_INST_FROM_L3	The processor's I-cache was reloaded from local core's L3 cache due to an instruction fetch (not prefetch).
PMC4	0000044044	PM_INST_FROM_L3.1_ECO_MOD	The processor's I-cache was reloaded with modified (M) data from another core's ECO L3 on the same chip due to an instruction fetch (not prefetch).
PMC4	0000044046	PM_INST_FROM_L2.1_MOD	The processor's I-cache was reloaded with modified (M) data from another core's L2 cache on the same chip due to an instruction fetch (not prefetch).

Table 5-1. IFU Events (Sheet 3 of 4)

PMC	Event Code	Event Name	Event Description
PMC4	0000044048	PM_INST_FROM_DL2L3_MOD	The processor's I-cache was reloaded with modified (M) data from another chip's L2 or L3 cache on a different node or group (distant) as this chip due to an instruction fetch (not prefetch).
PMC4	000004404A	PM_INST_FROM_OFF_CHIP_CACHE	The processor's I-cache was reloaded with either shared or modified data from another core's L2 or L3 cache on a different chip (remote or distant) due to an instruction fetch (not prefetch).
PMC4	000004404C	PM_INST_FROM_DMEM	The processor's I-cache was reloaded from another chip's memory on the same node or group (distant) due to an instruction fetch (not prefetch).
PMC4	000004404E	PM_INST_FROM_L3MISS_MOD	The processor's I-cache was reloaded from a location other than the local core's L3 cache due to an instruction fetch.
PMC4	0000044050	PM_INST_SYS_PUMP_MPRED_RTY	Final pump scope (system) ended up larger than the initial pump scope (chip/group) for an instruction fetch.
PMC4	0000044052	PM_INST_PUMP_MPRED	Pump misprediction. Counts across all types of pumps for an instruction fetch.
PMC4	00000400FA	PM_RUN_INST_CMPL	Run instructions.
Any PMC	0000004080	PM_INST_FROM_L1	Instruction fetches from the L1 cache. L1 instruction hit.
Any PMC	0000004880	PM_BANK_CONFLICT	Read blocked due to an interleave conflict. The IFAR logic will detect an interleave conflict and kill the data that was read that cycle.
Any PMC	0000004084	PM_EAT_FULL_CYC	Cycles, no room in EAT.
Any PMC	0000004884	PM_IBUF_FULL_CYC	Cycles, no room in IBUF.
Any PMC	0000004088	PM_IC_DEMAND_REQ	Demand instruction fetch request.
Any PMC	0000004888	PM_IC_PREF_REQ	Instruction prefetch request.
Any PMC	000000408C	PM_L1_DEMAND_WRITE	Instruction demand sectors written into the instruction L1 cache.
Any PMC	000000488C	PM_IC_PREF_WRITE	Instruction prefetch written into the instruction L1 cache.
Any PMC	0000004090	PM_IC_PREF_CANCEL_PAGE	Prefetch canceled due to a page boundary.
Any PMC	0000004890	PM_IC_PREF_CANCEL_HIT	Prefetch canceled due to an I-cache hit.
Any PMC	0000004094	PM_IC_PREF_CANCEL_L2	L2 squashed either a demand or prefetch request.
Any PMC	0000004894	PM_IC_RELOAD_PRIVATE	Reload the line that was brought in as private for a specific thread. Most lines are brought in shared for all eight threads. If RA does not match, then invalidate the line and bring it shared to the other thread.
Any PMC	0000004098	PM_IC_DEMAND_L2_BHT_REDIRECT	An I-cache demand request was sent to the L2 cache due to a BHT redirect.
Any PMC	0000004898	PM_IC_DEMAND_L2_BR_REDIRECT	An I-cache demand request was sent to the L2 cache due to a branch mispredict (15-cycle path).
Any PMC	00000048A8	PM_DECODE_FUSION_LD_ST_DISP	32-bit displacement D-form and 16-bit displacement X-form.
Any PMC	00000040BC	PM_THRD_PRIO_0_1_CYC	Number of cycles the thread is running at priority level 0 or 1.
Any PMC	00000048BC	PM_THRD_PRIO_2_3_CYC	Number of cycles the thread is running at priority level 2 or 3.
Any PMC	0000005080	PM_THRD_PRIO_4_5_CYC	Number of cycles the thread is running at priority level 4 or 5.

Table 5-1. IFU Events (Sheet 4 of 4)

PMC	Event Code	Event Name	Event Description
Any PMC	0000005880	PM_THRD_PRIO_6_7_CYC	Number of cycles the thread is running at priority level 6 or 7.
Any PMC	0000005888	PM_IC_INVALIDATE	Instruction cache line invalidated.
Any PMC	000000508C	PM_SHL_CREATED	Store-hit-load table entry created.
Any PMC	000000588C	PM_SHL_ST_DEP_CREATED	Store-hit-load table read hit with entry enabled.
Any PMC	0000005094	PM_IC_MISS_ICBI	Threaded version, I-cache misses with an EA directory hit but no sector valids are on. ICBI took the line out.
Any PMC	0000005894	PM_LWSYNC	An **lwsync** instruction was decoded and transferred.
Any PMC	000000589C	PM_PTESYNC	A **ptesync** instruction was counted when the instruction is decoded and transmitted.
Any PMC	00000050A0	PM_HWSYNC	A **hwsync** instruction was decoded and transferred.
Any PMC	00000058A4	PM_FLUSH_LSU	LSU flushes; includes all LSU flushes.
Any PMC	00000058A8	PM_DECODE_HOLD_ICT_FULL	Counts the number of cycles when the IFU was not able to decode and transmit one or more instructions because all itags were in use. This means the ICT is full for this thread.

5.2 Branch Events

Table 5-2 lists the branch events. Branch predictions and mispredictions from different predictors, such as the count cache, link stack, TAGE, BHT, are provided. Also provided are events that count taken versus not-taken branches.

Table 5-2. Branch Events (Sheet 1 of 3)

PMC	Event Code	Event Name	Event Description
PMC1	0000010068	PM_BRU_FIN	Branch instruction finished.
PMC2	0000020036	PM_BR_2PATH	Branches that are not strongly biased.
PMC2	0000020056	PM_TAKEN_BR_MPRED_CMPL	Total number of taken branches that were incorrectly predicted as not-taken. This event counts branches completed and does not include speculative instructions.
PMC2	000002505E	PM_BACK_BR_CMPL	Branch instruction completed with a target address less than the current instruction address.
PMC2	00000200FA	PM_BR_TAKEN_CMPL	New event for branch taken.
PMC3	000003005C	PM_BFU_BUSY	Cycles in which all four binary floating-point units (BFUs) are busy. The BFU is running at capacity.
PMC4	0000040036	PM_BR_2PATH	Branches that are not strongly biased.
PMC4	000004D05E	PM_BR_CMPL	Any branch instruction completed.
PMC4	00000400F6	PM_BR_MPRED_CMPL	Number of branch mispredicts.
Any PMC	000000409C	PM_BR_PRED	Conditional branch executed in which the hardware predicted the direction or target. Includes taken and not taken branches and is counted at execution time.
Any PMC	000000489C	PM_BR_CORECT_PRED_TAKEN_CMPL	Conditional branch completed in which the hardware correctly predicted the direction as taken. Counted at completion time.

Table 5-2. Branch Events (Sheet 2 of 3)

PMC	Event Code	Event Name	Event Description
Any PMC	00000040A0	PM_BR_UNCOND	Unconditional branch completed. Hardware branch prediction was not used for this branch. This can be an I-form branch, a B-form branch with BO-field set to branch always, or a B-form branch which was coverted to a resolve.
Any PMC	00000048A0	PM_BR_PRED_PCACHE	Conditional branch completed that used pattern cache prediction.
Any PMC	00000040A4	PM_BR_PRED_CCACHE	Conditional branch completed that used the count cache for target prediction.
Any PMC	00000048A4	PM_STOP_FETCH_ PENDING_CYC	Fetching is stopped due to an incoming instruction that will result in a flush.
Any PMC	00000040A8	PM_BR_PRED_LSTACK	Conditional branch completed that used the link stack for target prediction.
Any PMC	00000040AC	PM_BR_MPRED_ CCACHE	Conditional branch completed that was mispredicted due to the count cache target prediction.
Any PMC	00000048AC	PM_BR_MPRED_LSTACK	Conditional branch completed that was mispredicted due to the link stack target prediction.
Any PMC	00000040B0	PM_BR_PRED_TAKEN_ CR	Conditional branch that had its direction predicted. I-form branches do not set this event. In addition, B-form branches, which do not use the BHT, do not set this event. These are branches with the BO-field set to "always taken" and branches.
Any PMC	00000048B0	PM_BR_MPRED_ PCACHE	Conditional branch completed that was mispredicted due to pattern cache prediction.
Any PMC	00000040B4	PM_BR_PRED_TA	Conditional branch completed that had its target address predicted. Only XL-form branches set this event. This equals the sum of CCACHE, LSTACK, and PCACHE.
Any PMC	00000048B4	PM_DECODE_FUSION_ CONST_GEN	32-bit constant generation.
Any PMC	00000040B8	PM_BR_MPRED_TAKEN_ CR	A conditional branch that resolved to taken was mispredicted as not taken (due to the BHT direction prediction).
Any PMC	00000048B8	PM_BR_MPRED_TAKEN_ TA	Conditional branch completed that was mispredicted due to the target address prediction from the count cache or link stack. Only XL-form branches that resolved taken set this event.
Any PMC	0000005098	PM_LINK_STACK_ WRONG_ADD_PRED	Link stack predicts wrong address because of link-stack design limitation or software violating the coding conventions.
Any PMC	0000005898	PM_LINK_STACK_ INVALID_PTR	It is most often caused by certain types of flush where the pointer is not available. Can result in the data in the link stack becoming unusable.
Any PMC	00000058A0	PM_LINK_STACK_ CORRECT	Link stack predicts correct address.
Any PMC	00000050A4	PM_FLUSH_MPRED	Branch mispredict flushes. Includes target and address misprediction.
Any PMC	00000050A8	PM_EAT_FORCE_ MISPRED	XL-form branch was mispredicted due to the predicted target address missing from the effective address table (EAT). In this case, the EAT forces a mispredict because there is no predicated target to validate. This is a rare case that can occur when the EAT is full and a branch is issued.
Any PMC	00000050B0	PM_BTAC_BAD_RESULT	BTAC thinks the branch will be taken, but it is either predicted not-taken by the BHT or the target address is wrong (less common). In both cases, a redirect happens.
Any PMC	00000058B0	PM_BTAC_GOOD_ RESULT	BTAC predicts a taken branch, the BHT agrees, and the target address is correct.
Any PMC	00000050B4	PM_TAGE_CORRECT_ TAKEN_CMPL	The TAGE overwrites the BHT direction prediction and it was correct. Counted at completion for taken branches only.

Table 5-2. Branch Events (Sheet 3 of 3)

PMC	Event Code	Event Name	Event Description
Any PMC	00000058B4	PM_TAGE_CORRECT	The TAGE overwrites the BHT direction prediction and it was correct. Includes taken and not taken and is counted at execution time.
Any PMC	00000050B8	PM_TAGE_OVERRIDE_WRONG	The TAGE overwrites the BHT direction prediction but it was incorrect. Counted at completion for taken branches only.
Any PMC	00000058B8	PM_TAGE_OVERRIDE_WRONG_SPEC	The TAGE overwrites the BHT direction prediction and it was correct. Includes taken and not taken and is counted at execution time.

5.3 ISU Events

Table 5-3 lists the ISU events.

Table 5-3. ISU Events (Sheet 1 of 2)

PMC	Event Code	Event Name	Event Description
PMC1	0000010006	PM_DISP_HELD	Dispatch held.
PMC1	0000010028	PM_STALL_END_ICT_EMPTY	The number of times the core transitioned from a stall to ICT-empty for this thread.
PMC1	00000100F2	PM_1PLUS_PPC_CMPL	One or more PPC instructions finished.
PMC2	0000020006	PM_DISP_HELD_ISSQ_FULL	Dispatch held due to an issue queue full; includes issue queue and branch queue.
PMC2	0000020008	PM_ICT_EMPTY_CYC	Number of cycles in which the ICT is completely empty. No itags are assigned to any thread.
PMC2	000002001A	PM_NTC_ALL_FIN	Number of cycles after instruction finished to instruction completed.
PMC2	000002E016	PM_NTC_ISSUE_HELD_ARB	The next-to-complete (NTC) instruction is being held at dispatch because it lost arbitration onto the issue pipe to another instruction (from the same thread or a different thread).
PMC2	0000024050	PM_IOPS_CMPL	Internal operations completed.
PMC2	000002405A	PM_NTC_FIN	Number of cycles in which the oldest instruction in the pipeline (NTC) finishes. This event is used to account for cycles when work is being completed in the CPI stack.
PMC3	0000030008	PM_DISP_STARVED	Dispatched starved.
PMC3	0000030012	PM_FLUSH_COMPLETION	The instruction that was NTC did not complete because it suffered a flush.
PMC3	000003005A	PM_ISQ_0_8_ENTRIES	Cycles in which eight or less issue queue entries are in use. This is a shared event, not per thread.
PMC3	000003D05A	PM_NTC_ISSUE_HELD_OTHER	The NTC instruction is being held at dispatch during regular pipeline cycles, the VSU is busy with multi-cycle instructions, or a write-back collision occurred with the VSU.
PMC3	000003D05C	PM_DISP_HELD_HB_FULL	Dispatch held due to history buffer full. Could be GPR/VSR/VMR/FPR/CR/XVF; CR; XVF (XER/VSCR/FPSCR)
PMC4	000004000A	PM_ISQ_36_44_ENTRIES	Cycles in which 36 or more issue queue entries are in use. This is a shared event, not per thread. There are 44 issue queue entries across four slices in the whole core.
PMC4	00000400F2	PM_1PLUS_PPC_DISP	Cycles at least one instruction dispatched.
PMC4	00000400F8	PM_FLUSH	Flush (any type).

Table 5-3. ISU Events (Sheet 2 of 2)

PMC	Event Code	Event Name	Event Description
Any PMC	0000002080	PM_EE_OFF_EXT_INT	Cycles MSR[EE] is off and external interrupts are active.
Any PMC	0000002880	PM_FLUSH_DISP	Dispatch flush.
Any PMC	0000002084	PM_FLUSH_HB_ RESTORE_CYC	Cycles in which no new instructions can be dispatched to the ICT after a flush. History buffer recovery.
Any PMC	0000002884	PM_ISYNC	Isync completion count per thread.
Any PMC	0000002088	PM_FLUSH_DISP_SB	Dispatch flush: scoreboard.
Any PMC	0000002888	PM_FLUSH_DISP_ TLBIE	Dispatch flush: TLBIE.
Any PMC	000000208C	PM_CLB_HELD	Control logic block (CLB) indicates quadword fetch block. Hold: any reason.
Any PMC	000000288C	PM_DISP_CLB_HELD_ BAL	Dispatch/CLB hold: balance flush.
Any PMC	0000002090	PM_DISP_CLB_HELD_SB	Dispatch/CLB hold: scoreboard.
Any PMC	0000002890	PM_DISP_CLB_HELD_ TLBIE	Dispatch hold: due to TLBIE.
Any PMC	00000020B0	PM_LSU_FLUSH_NEXT	LSU flush next reported at flush time. Sometimes these also come with an exception.
Any PMC	00000028B0	PM_DISP_HELD_ TBEGIN	This outer **tbegin** transaction cannot be dispatched until the previous **tend** instruction completes.
Any PMC	0000003080	PM_ISU0_ISS_HOLD_ ALL	All ISU rejects.
Any PMC	0000003880	PM_ISU2_ISS_HOLD_ ALL	All ISU rejects.
Any PMC	0000003084	PM_ISU1_ISS_HOLD_ ALL	All ISU rejects.
Any PMC	0000003884	PM_ISU3_ISS_HOLD_ ALL	All ISU rejects.
PMC2	0000020058	PM_DARQ1_10_12_ ENTRIES	Cycles in which 10 or more DARQ1 entries (out of 12) are in use.
PMC2	000002005A	PM_DARQ1_7_9_ ENTRIES	Cycles in which 7 - 9 DARQ1 entries (out of 12) are in use.
PMC3	000003E050	PM_DARQ1_4_6_ ENTRIES	Cycles in which 4 - 6 DARQ1 entries (out of 12) are in use.
PMC4	000004C122	PM_DARQ1_0_3_ ENTRIES	Cycles in which three or fewer DARQ1 entries (out of 12) are in use.

5.4 VSU Events

Table 5-4 lists the VSU events.

Table 5-4. VSU Events

PMC	Event Code	Event Name	Event Description
PMC4	0000044054	PM_VECTOR_LD_ CMPL	Number of vector load instructions completed.
PMC4	0000044056	PM_VECTOR_ST_ CMPL	Number of vector store instructions completed.
PMC2	000002000E	PM_FXU_BUSY	Cycles in which all four FXUs are busy. The FXU is running at capacity.
PMC2	0000024052	PM_FXU_IDLE	Cycles in which FXU0, FXU1, FXU2, and FXU3 are all idle.
PMC2	000002505C	PM_VSU_FIN	VSU instruction finished. Up to four per cycle.
PMC3	000003000E	PM_FXU_1PLUS_ BUSY	At least one of the four FXU units is busy.
PMC3	000003D058	PM_VSU_DP_FSQRT_ FDIV	Vector versions of **fdiv**, **fsqrt**
PMC4	0000040004	PM_FXU_FIN	The fixed-point unit finished an instruction. Instructions that finish might not necessarily complete.
PMC4	000004D04C	PM_DFU_BUSY	Cycles in which all four decimal floating-point units (DFU) are busy. The DFU is running at capacity.
PMC4	000004D04E	PM_VSU_FSQRT_ FDIV	4-FLOP instruction (**fdiv**, **fsqrt**). Scalar instructions only.
PMC4	000004D050	PM_VSU_NON_FLOP_ CMPL	Non-FLOP instruction completed.
PMC4	000004D052	PM_2FLOP_CMPL	DP vector version of **fmul**, **fsub**, **fcmp**, **fsel**, **fabs**, **fnabs**, **fres**, **fsqrte**, **fneg**
PMC4	000004D054	PM_8FLOP_CMPL	8-FLOP instruction completed.
PMC4	000004D056	PM_NON_FMA_FLOP_ CMPL	Non-FMA instruction completed.
PMC4	000004D058	PM_VECTOR_FLOP_ CMPL	Vector FP instruction completed.
PMC4	000004D05A	PM_NON_MATH_ FLOP_CMPL	Non-FLOP instruction completed.
PMC4	000004D05C	PM_DP_QP_FLOP_ CMPL	Double-precision or quad-precision instruction completed.
PMC4	0000045050	PM_1FLOP_CMPL	One FLOP (**fadd**, **fmul**, **fsub**, **fcmp**, **fsel**, **fabs**, **fnabs**, **fres**, **fsqrte**, **fneg**) instruction completed.
PMC4	0000045052	PM_4FLOP_CMPL	4-FLOP instruction completed.
PMC4	0000045054	PM_FMA_CMPL	Two-flops instruction completed (**fmadd**, **fnmadd**, **fmsub**, **fnmsub**). Scalar instructions only.
PMC4	0000045056	PM_SCALAR_FLOP_ CMPL	Scalar flop operation completed.
PMC4	0000045058	PM_IC_MISS_CMPL	Non-speculative I-cache miss. Counted at completion.
PMC4	000004505A	PM_SP_FLOP_CMPL	SP instruction completed.
PMC4	000004505C	PM_MATH_FLOP_ CMPL	Math FLOP instruction completed.
PMC4	000004505E	PM_FLOP_CMPL	Floating-point operation finished.

5.5 LSU Events

Table 5-5 lists the LSU events.

Table 5-5. LSU Events (Sheet 1 of 8)

PMC	Event Code	Event Name	Event Description
PMC1	000001001A	PM_LSU_SRQ_FULL_CYC	Cycles in which the store queue is full on all four slices. This is event is not per thread. All the threads see the same count for this core resource.
PMC1	000001002C	PM_L1_DCACHE_RELOADED_ALL	L1 data cache reloaded for demand. If MMCR1[16] is '1', prefetches are also included.
PMC1	000001002E	PM_LMQ_MERGE	A demand miss collides with a prefetch for the same line.
PMC1	0000010050	PM_CHIP_PUMP_CPRED	Initial and final pump scope was chip pump (prediction is correct) for all data types excluding data prefetch (demand load, instruction prefetch, instruction fetch, xlate).
PMC1	0000010052	PM_GRP_PUMP_MPRED_RTY	Final pump scope (group) ended up larger than initial pump scope (chip) for all data types excluding data prefetch (demand load, instruction prefetch, instruction fetch, xlate).
PMC1	0000010054	PM_PUMP_CPRED	Pump prediction correct. Counts across all types of pumps for all data types excluding data prefetch (demand load, instruction prefetch, instruction fetch, xlate).
PMC1	0000010056	PM_MEM_READ	Reads from memory from this thread (includes data, instructions, xlate, L1 prefetch, instruction prefetch); also includes the L4 cache.
PMC1	0000010058	PM_MEM_LOC_THRESH_IFU	Local memory above threshold for IFU speculation control.
PMC1	000001C05E	PM_MEM_LOC_THRESH_LSU_MED	Local memory above threshold for data prefetch.
PMC1	000001D058	PM_DARQ0_10_12_ENTRIES	Cycles in which 10 or more DARQ entries (out of 12) are in use.
PMC1	000001E058	PM_STCX_FAIL	A **stcx** instruction failed.
PMC1	0000010062	PM_LD_L3MISS_PEND_CYC	Number of cycles that an L3 miss was pending for this thread.
PMC1	000001006A	PM_NTC_ISSUE_HELD_DARQ_FULL	The NTC instruction is being held at dispatch because there are no slots in the DARQ for it.
PMC1	000001006C	PM_RUN_CYC_ST_MODE	Cycles run latch is set and core is in ST mode.
PMC1	00000100FC	PM_LD_REF_L1	All L1 D-cache load references counted at finish, gated by reject.
PMC2	0000020016	PM_ST_FIN	Store finish count. Includes speculative activity.
PMC2	0000020018	PM_ST_FWD	Store forwards that finished.
PMC2	000002E014	PM_STCX_FIN	Number of **stcx** instructions finished. This includes instructions in the speculative path of a branch that might be flushed.
PMC2	000002003E	PM_LSU_LMQ_SRQ_EMPTY_CYC	Cycles in which the LSU is empty for all threads (LMQ and SRQ are completely empty).
PMC2	000002C04E	PM_LD_MISS_L1_FIN	Number of load instructions that finished with an L1 miss. Note that even if a load spans multiple slices, this event increments only once per load operation.
PMC2	0000020050	PM_GRP_PUMP_CPRED	Initial and final pump scope and data sourced across this scope was group pump for all data types excluding data prefetch (demand load, instruction prefetch, instruction fetch, xlate).

Table 5-5. LSU Events (Sheet 2 of 8)

PMC	Event Code	Event Name	Event Description
PMC2	0000020052	PM_GRP_PUMP_MPRED	Final pump scope (group) ended up either larger or smaller than initial pump scope for all data types excluding data prefetch (demand load, instruction prefetch, instruction fetch, xlate).
PMC2	000002C058	PM_MEM_PREF	Memory prefetch for this thread includes the L4 cache.
PMC2	000002E050	PM_DARQ0_7_9_ ENTRIES	Number of cycles in which 7, 8, or 9 DARQ entries (out of 12) are in use.
PMC2	000002E05A	PM_LRQ_REJECT	Internal LSU reject from LRQ. Rejects cause the load to go back to LRQ, but it stays contained within the LSU once it is issued. This event counts the number of times the LRQ attempts to relaunch an instruction after a reject. Any load can suffer multiple rejects.
PMC2	000002E05C	PM_LSU_REJECT_ERAT _MISS	LSU reject due to an ERAT (up to four per cycle).
PMC2	000002E05E	PM_LMQ_EMPTY_CYC	Number of cycles in which the LMQ has no pending load misses for this thread.
PMC2	00000200F0	PM_ST_CMPL	Stores completed from second-level store queue (S2Q).
PMC3	000003001C	PM_LSU_REJECT_LMQ_ FULL	LSU reject due to an LMQ full (up to four per cycle).
PMC3	000003504E	PM_DARQ0_4_6_ ENTRIES	Number of cycles in which 4, 5, or 6 DARQ entries (out of 12) are in use.
PMC3	0000030050	PM_SYS_PUMP_CPRED	Initial and final pump scope was system pump for all data types excluding data prefetch (demand load, instruction prefetch, instruction fetch, xlate).
PMC3	0000030052	PM_SYS_PUMP_MPRED	Final pump scope (system) mispredicted. Either the original scope was too small (chip/group) or the original scope was system and it should have been smaller. Counts for all data types excluding data prefetch (demand load, instruction prefetch, instruction fetch, xlate).
PMC3	0000030058	PM_TLBIE_FIN	A **tlbie** instruction finished.
PMC3	000003C058	PM_LARX_FIN	An **larx** instruction finished.
PMC3	000003C05E	PM_MEM_RWITM	Memory read with intent to modify for this thread.
PMC3	0000034054	PM_PARTIAL_ST_FIN	Any store finished by an LSU slice.
PMC3	000003E054	PM_LD_MISS_L1	Load missed L1 cache, counted at execution time (can be greater than loads finished). LMQ merges are not included in this count. For example, if a load instruction misses on an address that is already allocated on the LMQ, this event will not increment for that load. Note that this count is per slice. If a load spans multiple slices, this event will increment multiple times for a single load.
PMC3	0000030064	PM_DARQ_STORE_XMIT	The DARQ attempted to transmit a store into an LSAQ or SRQ entry (includes rejects). Not qualified by thread, so it includes counts for the whole core.
PMC3	0000030066	PM_LSU_FIN	LSU finished a PPC instruction (up to four per cycle).
PMC3	00000300F0	PM_ST_MISS_L1	Store missed the L1 cache.
PMC3	00000300F6	PM_L1_DCACHE_ RELOAD_VALID	Data L1 cache reloaded due to a demand load.
PMC3	00000300FC	PM_DTLB_MISS	Data PTEG reload.
PMC4	0000040008	PM_SRQ_EMPTY_CYC	Number of cycles in which the SRQ has at least one (out of four) empty slice.
PMC4	000004003E	PM_LD_CMPL	Count of loads completed.
PMC4	000004D04A	PM_DARQ0_0_3_ ENTRIES	Number of cycles in which three or less DARQ entries (out of 12) are in use.

Table 5-5. LSU Events (Sheet 3 of 8)

PMC	Event Code	Event Name	Event Description
PMC4	0000040050	PM_SYS_PUMP_MPRED_RTY	Final pump scope (system) ended up larger than Initial pump scope (chip/group) for all data types excluding data prefetch (demand load, instruction prefetch, instruction fetch, xlate).
PMC4	0000040052	PM_PUMP_MPRED	Pump misprediction. Counts across all types of pumps for all data types excluding data prefetch (demand load, instruction prefetch, instruction fetch, xlate).
PMC4	0000040056	PM_MEM_LOC_THRESH_LSU_HIGH	Local memory above threshold for LSU medium.
PMC4	000004C058	PM_MEM_CO	Memory castouts from this thread.
PMC4	000004405E	PM_DARQ_STORE_REJECT	The DARQ attempted to transmit a store into an LSAQ or SRQ entry but It was rejected. Divide by PM_DARQ_STORE_XMIT to get reject ratio.
PMC4	000004E05C	PM_LSU_REJECT_LHS	LSU reject due to LHS (up to four per cycle).
PMC4	00000400F0	PM_LD_MISS_L1	Load missed L1 cache, at execution time (not gated by finish, which means this counter can be greater than loads finished).
Any PMC	000000C080	PM_LS0_LD_VECTOR_FIN	LS0 finished load vector operation.
Any PMC	000000C880	PM_LS1_LD_VECTOR_FIN	LS1 finished load vector operation.
Any PMC	000000C084	PM_LS2_LD_VECTOR_FIN	LS2 finished load vector operation.
Any PMC	000000C884	PM_LS3_LD_VECTOR_FIN	LS3 finished load vector operation.
Any PMC	000000C088	PM_LSU_DTLB_MISS_4K	Data TLB miss page size 4 KB.
Any PMC	000000C888	PM_LSU_DTLB_MISS_64K	Data TLB miss page size 64 KB.
Any PMC	000000C08C	PM_LSU_DTLB_MISS_16M_2M	Data TLB miss page size 16 MB (HPT) or 2 MB (Radix).
Any PMC	000000C88C	PM_LSU_DTLB_MISS_16G_1G	Data TLB miss page size 16 GB (HPT) or 1 GB (Radix).
Any PMC	000000C090	PM_LSU_STCX	A **stcx** instruction is executed and sent to the nest.
Any PMC	000000C890	PM_LSU_NCST	Non-cacheable stores sent to nest.
Any PMC	000000C094	PM_LS0_UNALIGNED_LD	Load instructions whose data crosses a double-word boundary, which causes it to require an additional slice than what normally would be required of the load of that size. If the load wraps from slice 3 to slice 0, there is an additional 3-cycle penalty.
Any PMC	000000C894	PM_LS1_UNALIGNED_LD	Load instructions whose data crosses a double-word boundary, which causes it to require an additional slice than what normally would be required of the load of that size. If the load wraps from slice 3 to slice 0, thee is an additional 3-cycle penalty.
Any PMC	000000C098	PM_LS2_UNALIGNED_LD	Load instructions whose data crosses a double-word boundary, which causes it to require an additional slice than what normally would be required of the load of that size. If the load wraps from slice 3 to slice 0, thee is an additional 3-cycle penalty.
Any PMC	000000C898	PM_LS3_UNALIGNED_LD	Load instructions whose data crosses a double-word boundary, which causes it to require an additional slice than what normally would be required of the load of that size. If the load wraps from slice 3 to slice 0, thee is an additional 3-cycle penalty.
Any PMC	000000C09C	PM_LS0_LAUNCH_HELD_PREF	Number of times a load or store instruction was unable to launch/relaunch because a high-priority prefetch used that relaunch the cycle.

Table 5-5. LSU Events (Sheet 4 of 8)

PMC	Event Code	Event Name	Event Description
Any PMC	000000C89C	PM_LS1_LAUNCH_HELD_PREF	Number of times a load or store instruction was unable to launch/relaunch because a high-priority prefetch used that relaunch cycle.
Any PMC	000000C0A0	PM_LSU0_FALSE_LHS	False LHS match detected.
Any PMC	000000C8A0	PM_LSU1_FALSE_LHS	False LHS match detected.
Any PMC	000000C0A4	PM_LSU2_FALSE_LHS	False LHS match detected.
Any PMC	000000C8A4	PM_LSU3_FALSE_LHS	False LHS match detected.
Any PMC	000000C0A8	PM_LSU_FLUSH_CI	Load was not issued to the LSU as a cache inhibited (non-cacheable) load, but it was later determined to be cache inhibited.
Any PMC	000000C8A8	PM_LSU_FLUSH_ATOMIC	Quadword loads (**lq**) are considered atomic because they always span at least two slices. If a snoop or store from another thread changes the data, the load is accessing between two or three pieces of the **lq** instruction. The **lq** instruction is flushed.
Any PMC	000000C0AC	PM_LSU_FLUSH_EMSH	An ERAT miss was detected after a set-p hit. The ERAT tracker indicates a fail due to a **tlbmiss** and the instruction is flushed because the instruction was working on the wrong address.
Any PMC	000000C8AC	PM_LSU_FLUSH_RELAUNCH_MISS	If a load that has already returned data and has to relaunched for any reason and then gets a miss (ERAT, set-p, data cache), it will often be flushed at relaunch time because the data might be inconsistent.
Any PMC	000000C0B0	PM_LSU_FLUSH_UE	Correctable ECC error on reload data, reported at critical data forward time.
Any PMC	000000C8B0	PM_LSU_FLUSH_LHS	Effective address alias flush: no EA match but a real address match. If the data has not yet been returned for this load, the instruction is rejected. If the data has been returned for this load, it is flushed.
Any PMC	000000C0B4	PM_LSU_FLUSH_WRK_ARND	LSU workaround flush. These flushes are set up with programmable scan-only latches to perform various actions when the flush macro receives a trigger from the debug macros.These actions include things like flushing the next operation encountered for a particular thread or flushing the next operation that is an NTC operation that is encountered on a particular slice.The type of flush that the workaround is set up to perform is highly variable.
Any PMC	000000C8B4	PM_LSU_FLUSH_LHL_SHL	The instruction was flushed because of a sequential load/store consistency issue, because a load or store hits on an older load that has either been snooped (for loads) or has stale data (for stores).
Any PMC	000000C0B8	PM_LSU_FLUSH_SAO	A load-hit-load condition with strong address ordering (SAO) will have address compare disabled and flushed.
Any PMC	000000C8B8	PM_LSU_FLUSH_LARX_STCX	A **larx** is flushed because an older **larx** has an LMQ reservation for the same thread. A **stcx** is flushed because an older **stcx** is in the LMQ. The flush happens when the older **larx**/**stcx** relaunches.
Any PMC	000000C0BC	PM_LSU_FLUSH_OTHER	Other LSU flushes including: **sync** (**sync** acknowledged from the L2 cache caused a search of the LRQ for the oldest snooped load. This either signals a precise flush of the oldest snooped load or a flush the next PPC instruction); data valid flush next (one example is a store and reload are lined up such that a store-hit-reload scenario exists and the CDF has already launched and has received bad/stale data); bad data valid flush next [for example, a **larx** return data and dval but cannot allocate to LMQ (because LMQ is full or another reason). Already gave dval but cannot watch it for snoop_hit_larx. Must take the "bad dval" back and flush all younger operations].
Any PMC	000000C8BC	PM_STCX_SUCCESS_CMPL	Number of **stcx** instructions that completed successfully.
Any PMC	000000D080	PM_LSU0_SET_MPRED	Set prediction (set-p) miss. The entry was not found in the set-prediction table.
Any PMC	000000D880	PM_LSU1_SET_MPRED	Set prediction (set-p) miss. The entry was not found in the set-prediction table.
Any PMC	000000D084	PM_LSU2_SET_MPRED	Set prediction (set-p) miss. The entry was not found in the set-prediction table.

Table 5-5. LSU Events (Sheet 5 of 8)

PMC	Event Code	Event Name	Event Description
Any PMC	000000D884	PM_LSU3_SET_MPRED	Set prediction (set-p) miss. The entry was not found in the set-prediction table.
Any PMC	000000D088	PM_LSU0_LDMX_FIN	The thread has executed an **ldmx** instruction that accessed a doubleword that contains an effective address within an enabled section of the load monitored region. Therefore, this event should not occur if FSCR has disabled the load monitored facility (FSCR[52]) or disabled the EBB facility (FSCR[56]). **Note:** This event should only be asserted on the master slice finish because the PPC finish count is required.
Any PMC	000000D888	PM_LSU1_LDMX_FIN	The thread has executed an **ldmx** instruction that accessed a doubleword that contains an effective address within an enabled section of the load monitored region. Therefore, this event should not occur if FSCR has disabled the load monitored facility (FSCR[52]) or disabled the EBB facility (FSCR[56]).
Any PMC	000000D08C	PM_LSU2_LDMX_FIN	The thread has executed an **ldmx** instruction that accessed a doubleword that contains an effective address within an enabled section of the load monitored region. Therefore, this event should not occur if FSCR has disabled the load monitored facility (FSCR[52]) or disabled the EBB facility (FSCR[56]).
Any PMC	000000D88C	PM_LSU3_LDMX_FIN	The thread has executed an **ldmx** instruction that accessed a doubleword that contains an effective address within an enabled section of the load monitored region. Therefore, this event should not occur if FSCR has disabled the load monitored facility (FSCR[52]) or disabled the EBB facility (FSCR[56]).
Any PMC	000000D090	PM_LS0_DC_ COLLISIONS	Read-write data cache collisions.
Any PMC	000000D890	PM_LS1_DC_ COLLISIONS	Read-write data cache collisions.
Any PMC	000000D094	PM_LS2_DC_ COLLISIONS	Read-write data cache collisions.
Any PMC	000000D894	PM_LS3_DC_ COLLISIONS	Read-write data cache collisions.
Any PMC	000000D198	PM_LSU_FLUSH_ ATOMIC	Quadword loads (**lq**) are considered atomic because they always span at least two slices. If a snoop or store from another thread changes the data, the load is accessing between the second or third pieces of the **lq** instruction and the lq is flushed.
Any PMC	000000D998	PM_LSU_FLUSH_EMSH	An ERAT miss was detected after a set-p hit. The ERAT tracker indicates a fail due to a **tlbmiss**. The instruction is flushed, because the instruction was working on the wrong address.
Any PMC	000000D19C	PM_LSU_FLUSH_ RELAUNCH_MISS	If a load that has already returned data and has to relaunch for any reason, it gets a miss (ERAT, set-p, data cache). It will often be flushed at relaunch time because the data might be inconsistent.
Any PMC	000000D99C	PM_LSU_FLUSH_UE	Correctable ECC error on reload data, reported at critical data forward time.
Any PMC	000000D1A0	PM_LSU_FLUSH_LHS	Effective address alias flush. No EA match but a real address match. If the data has not yet been returned for this load, the instruction is rejected. However, if the data has ben returned, it is flushed.
Any PMC	000000D9A0	PM_LSU_FLUSH_LHL_ SHL	The instruction was flushed because of a sequential load/store consistency. If a load or store hits on an older load that has either been snooped (for loads) or has stale data (for stores).
Any PMC	000000D1A4	PM_LSU_FLUSH_SAO	A load-hit-load condition with strong address ordering has address compare disabled and flushed.
Any PMC	000000D9A4	PM_LSU_FLUSH_LARX_ STCX	A **larx** is flushed because an older **larx** has an LMQ reservation for the same thread. A **stcx** is flushed because an older **stcx** is in the LMQ. The flush happens when the older **larx**/**stcx** relaunches.
Any PMC	000000D0AC	PM_SRQ_SYNC_CYC	A **sync** is in the S2Q (edge detect to count).

Table 5-5. LSU Events (Sheet 6 of 8)

PMC	Event Code	Event Name	Event Description
Any PMC	000000D0B4	PM_LSU0_SRQ_S0_ VALID_CYC	Per thread, use edge detect to count allocates. On a per thread basis, level signal indicating Slot 0 is valid. By instrumenting a single slot, service time for that slot can be calculated. Previous machines required a separate signal indicating the slot was allocated.
Any PMC	000000D8B4	PM_LSU0_LRQ_S0_ VALID_CYC	Per thread, use edge detect to count allocates. On a per thread basis, level signal indicating Slot 0 is valid. By instrumenting a single slot, service time for that slot can be calculated. Previous machines required a separate signal indicating the slot was allocated.
Any PMC	000000D0B8	PM_LSU_LMQ_FULL_ CYC	Shared, not per thread.
Any PMC	000000D8B8	PM_LSU0_LMQ_S0_ VALID	Per thread, use edge detect to count allocates On a per thread basis, level signal indicating Slot 0 is valid. By instrumenting a single slot, service time for that slot can be calculated. Previous machines required a separate signal indicating the slot was allocated.
Any PMC	000000D0BC	PM_LSU0_1_LRQF_FULL _CYC	Counts the number of cycles the LRQF is full. LRQF is the queue that holds loads between finish and completion. If it fills up, instructions stay in LRQ until completion, potentially backing up the LRQ.
Any PMC	000000D8BC	PM_LSU2_3_LRQF_FULL _CYC	Counts the number of cycles the LRQF is full. LRQF is the queue that holds loads between finish and completion. If it fills up, instructions stay in LRQ until completion, potentially backing up the LRQ.
Any PMC	000000E080	PM_S2Q_FULL	Number of cycles during which the S2Q is full.
Any PMC	000000E880	PM_L1_SW_PREF	Software L1 prefetches, including software transient prefetches.
Any PMC	000000E084	PM_LS0_ERAT_MISS_ PREF	LS0 ERAT miss due to a prefetch.
Any PMC	000000E884	PM_LS1_ERAT_MISS_ PREF	LS1 ERAT miss due to a prefetch.
Any PMC	000000E088	PM_LS2_ERAT_MISS_ PREF	LS0 ERAT miss due to a prefetch.
Any PMC	000000E888	PM_LS3_ERAT_MISS_ PREF	LS1 ERAT miss due to a prefetch.
Any PMC	000000E08C	PM_LSU0_ERAT_HIT	Primary ERAT hit. There is no secondary ERAT.
Any PMC	000000E88C	PM_LSU1_ERAT_HIT	Primary ERAT hit. There is no secondary ERAT.
Any PMC	000000E090	PM_LSU2_ERAT_HIT	Primary ERAT hit. There is no secondary ERAT.
Any PMC	000000E890	PM_LSU3_ERAT_HIT	Primary ERAT hit. There is no secondary ERAT.
Any PMC	000000E094	PM_LSU0_TM_L1_HIT	Load TM hit in L1.
Any PMC	000000E894	PM_LSU1_TM_L1_HIT	Load TM hit in L1.
Any PMC	000000E098	PM_LSU2_TM_L1_HIT	Load TM hit in L1.
Any PMC	000000E898	PM_LSU3_TM_L1_HIT	Load TM hit in L1.
Any PMC	000000E09C	PM_LSU0_TM_L1_MISS	Load TM L1 miss.
Any PMC	000000E89C	PM_LSU1_TM_L1_MISS	Load TM L1 miss.
Any PMC	000000E0A0	PM_LSU2_TM_L1_MISS	Load TM L1 miss.
Any PMC	000000E8A0	PM_LSU3_TM_L1_MISS	Load TM L1 miss.
Any PMC	000000F080	PM_LSU_STCX_FAIL	The LSU detects the condition that a **stcx** instruction failed. No requirement to wait for a response from the nest.
Any PMC	000000F880	PM_SNOOP_TLBIE	TLBIE snoop.

Table 5-5. LSU Events (Sheet 7 of 8)

PMC	Event Code	Event Name	Event Description
Any PMC	000000F084	PM_PTE_PREFETCH	PTE prefetches.
Any PMC	000000F088	PM_LSU0_STORE_ REJECT	LSU0 store reject. All internal store rejects cause the instruction to go back to the SRQ and go to sleep until woken up to try again after the condition has been met.
Any PMC	000000F888	PM_LSU1_STORE_ REJECT	LSU1 store reject. All internal store rejects cause the instruction to go back to the SRQ and go to sleep until woken up to try again after the condition has been met.
Any PMC	000000F08C	PM_LSU2_STORE_ REJECT	LSU2 store reject. All internal store rejects cause the instruction to go back to the SRQ and go to sleep until woken up to try again after the condition has been met.
Any PMC	000000F88C	PM_LSU3_STORE_ REJECT	LSU3 store reject. All internal store rejects cause the instruction to go back to the SRQ and go to sleep until woken up to try again after the condition has been met.
Any PMC	000000F090	PM_LSU0_L1_CAM_ CANCEL	LSU0 L1 TM CAM cancel.
Any PMC	000000F890	PM_LSU1_L1_CAM_ CANCEL	LSU1 L1 TM CAM cancel.
Any PMC	000000F094	PM_LSU2_L1_CAM_ CANCEL	LSU2 L1 TM CAM cancel.
Any PMC	000000F894	PM_LSU3_L1_CAM_ CANCEL	LSU3 L1 TM CAM cancel.
Any PMC	000000F0A0	PM_DATA_STORE	All operations that drain from S2Q to L2 containing data.
Any PMC	000000F8A0	PM_NON_DATA_STORE	All operations that drain from S2Q to L2 and do not contain data.
Any PMC	000000F0A4	PM_DC_PREF_HW_ ALLOC	Prefetch stream allocated by the hardware prefetch mechanism.
Any PMC	000000F8A4	PM_DC_PREF_SW_ ALLOC	Prefetch stream allocated by software prefetching.
Any PMC	000000F0A8	PM_DC_PREF_CONF	A demand load referenced a line in an active prefetch stream. The stream can be allocated through the hardware prefetch mechanism or through software. It includes forward and backward streams.
Any PMC	000000F8A8	PM_DC_PREF_FUZZY_ CONF	A demand load referenced a line in an active fuzzy prefetch stream. The stream can be allocated through the hardware prefetch mechanism or through software. This event counts the number of fuzzy stream confirms (out-of-order effects or prefetch cannot keep up).
Any PMC	000000F0AC	PM_DC_PREF_STRIDED _CONF	A demand load referenced a line in an active strided prefetch stream. The stream can be allocated through the hardware prefetch mechanism or through software.
Any PMC	000000F8AC	PM_DC_DEALLOC_NO_ CONF	A demand load referenced a line in an active fuzzy prefetch stream. The stream can be allocated through the hardware prefetch mechanism or through software. This event counts the number of fuzzy stream confirms (out-of-order effects or prefetch cannot keep up).
Any PMC	000000F0B0	PM_L3_LD_PREF	An L3 load prefetch, sourced from a hardware or software stream, is sent to the nest.
Any PMC	000000F8B0	PM_L3_SW_PREF	An L3 load prefetch, sourced from a software prefetch stream, is sent to the nest.
Any PMC	000000F0B4	PM_DC_PREF_CONS_ ALLOC	The sum of this pair subtracted from the total number of allocates gives the total allocates in the normal phase.
Any PMC	000000F8B4	PM_DC_PREF_XCONS_ ALLOC	Prefetch stream allocated in the ultra conservative phase by either the hardware prefetch mechanism or software prefetch.

Table 5-5. LSU Events (Sheet 8 of 8)

PMC	Event Code	Event Name	Event Description
Any PMC	000000F0B8	PM_LS0_UNALIGNED_ST	Store instructions whose data crosses a double-word boundary, which causes it to require an additional slice than what normally would be required of the store of that size. If the store wraps from slice 3 to slice 0, there is an additional 3-cycle penalty.
Any PMC	000000F8B8	PM_LS1_UNALIGNED_ST	Store instructions whose data crosses a double-word boundary, which causes it to require an additional slice than what normally would be required of the store of that size. If the store wraps from slice 3 to slice 0, there is an additional 3-cycle penalty.
Any PMC	000000F0BC	PM_LS2_UNALIGNED_ST	Store instructions whose data a double-word boundary, which causes it to require an additional slice than what normally would be required of the store of that size. If the store wraps from slice 3 to slice 0, there is an additional 3-cycle penalty.
Any PMC	000000F8BC	PM_LS3_UNALIGNED_ST	Store instructions whose data crosses a double-word boundary, which causes it to require an additional slice than what normally would be required of the store of that size. If the store wraps from slice 3 to slice 0, there is an additional 3-cycle penalty.
PMC2	0000020054	PM_L1_PREF	A data line was written to the L1 cache due to a hardware or software prefetch.

5.6 Data Source Events

Table 5-6 lists the data source events.

Table 5-6. Data Source Events (Sheet 1 of 3)

PMC	Event Code	Event Name	Event Description
PMC1	000001C040	PM_DATA_FROM_L2_NO_CONFLICT	The processor's data cache was reloaded from the local core's L2 cache without conflict due to a demand load.
PMC1	000001C042	PM_DATA_FROM_L2	The processor's data cache was reloaded from the local core's L2 due to a demand load.
PMC1	000001C044	PM_DATA_FROM_L3_NO_CONFLICT	The processor's data cache was reloaded from local core's L3 cache without conflict due to a demand load.
PMC1	000001C046	PM_DATA_FROM_L3.1_SHR	The processor's data cache was reloaded with shared (S) data from another core's L3 cache on the same chip due to a demand load.
PMC1	000001C048	PM_DATA_FROM_ON_CHIP_CACHE	The processor's data cache was reloaded either shared or modified (M) data from another core's L or L3 cache on the same chip due to a demand load.
PMC1	000001C04A	PM_DATA_FROM_RL2L3_SHR	The processor's data cache was reloaded with shared data from another chip's L2 or L3 cache on the same node or group (remote) from this chip due to a demand load.
PMC1	000001C04C	PM_DATA_FROM_LL4	The processor's data cache was reloaded from the local chip's L4 cache due to a demand load.
PMC1	000001C04E	PM_DATA_FROM_L2MISS_MOD	The processor's data cache was reloaded from a location other than the local core's L2 cache due to a demand load.
PMC1	000001C050	PM_DATA_CHIP_PUMP_CPRED	Initial and final pump scope was chip pump (prediction is correct) for a demand load.
PMC1	000001C052	PM_DATA_GRP_PUMP_MPRED_RTY	Final pump scope (group) ended up larger than initial pump scope (chip) for a demand load.
PMC1	000001C054	PM_DATA_PUMP_CPRED	Pump prediction correct. Counts across all types of pumps for a demand load.

Table 5-6. Data Source Events (Sheet 2 of 3)

PMC	Event Code	Event Name	Event Description
PMC2	000002C040	PM_DATA_FROM_L2_MEPF	The processor's data cache was reloaded from local core's L2 hit without dispatch conflicts on Mepf state due to a demand load.
PMC2	000002C042	PM_DATA_FROM_L3_MEPF	The processor's data cache was reloaded from local core's L3 hit without dispatch conflicts hit on Mepf state due to a demand load.
PMC2	000002C044	PM_DATA_FROM_L3.1_MOD	The processor's data cache was reloaded with modified data from another core's L3 cache on the same chip due to a demand load.
PMC2	000002C046	PM_DATA_FROM_RL2L3_MOD	The processor's data cache was reloaded with modified data from another chip's L2 or L3 cache on the same node or group (remote), from this chip due to a demand load.
PMC2	000002C048	PM_DATA_FROM_LMEM	The processor's data cache was reloaded from the local chip's memory due to a demand load.
PMC2	000002C04A	PM_DATA_FROM_RL4	The processor's data cache was reloaded from another chip's L4 cache on the same node or group (remote) due to a demand load.
PMC2	000002C050	PM_DATA_GRP_PUMP_CPRED	Initial and final pump scope was group pump (prediction is correct) for a demand load.
PMC2	000002C052	PM_DATA_GRP_PUMP_MPRED	Final pump scope (group) ended up either larger or smaller than the initial pump scope for a demand load.
PMC2	00000200FE	PM_DATA_FROM_L2MISS	Demand load, L2 miss (not L2 hit).
PMC3	000003001A	PM_DATA_TABLEWALK_CYC	Data tablewalk cycles. Could be one or two active tablewalks; includes data prefetches.
PMC3	000003C040	PM_DATA_FROM_L2_DISP_CONFLICT_LDHITST	The processor's data cache was reloaded from the local core's L2 cache with load-hit-store conflict due to a demand load.
PMC3	000003C042	PM_DATA_FROM_L3_DISP_CONFLICT	The processor's data cache was reloaded from the local core's L3 cache with dispatch conflict due to a demand load.
PMC3	000003C044	PM_DATA_FROM_L3.1_ECO_SHR	The processor's data cache was reloaded with shared data from another core's ECO L3 cache on the same chip due to a demand load.
PMC3	000003C046	PM_DATA_FROM_L2.1_SHR	The processor's data cache was reloaded with shared data from another core's L2 cache on the same chip due to a demand load.
PMC3	000003C048	PM_DATA_FROM_DL2L3_SHR	The processor's data cache was reloaded with shared data from another chip's L2 or L3 cache on a different node or group (distant), from this chip due to a demand load.
PMC3	000003C04A	PM_DATA_FROM_RMEM	The processor's data cache was reloaded from another chip's memory on the same node or group (remote) due to a demand load.
PMC3	000003C04C	PM_DATA_FROM_DL4	The processor's data cache was reloaded from another chip's L4 cache on a different node or group (distant) due to a demand load.
PMC3	000003C050	PM_DATA_SYS_PUMP_CPRED	Initial and final pump scope was system pump (prediction is correct) for a demand load.
PMC3	000003C052	PM_DATA_SYS_PUMP_MPRED	Final pump scope (system) mispredicted. Either the original scope was too small (chip/group) or the original scope was system and it should have been smaller. Counts for a demand load.
PMC3	00000300FE	PM_DATA_FROM_L3MISS	Demand load, L3 miss (not L2 hit and not L3 hit).
PMC4	000004C040	PM_DATA_FROM_L2_DISP_CONFLICT_OTHER	The processor's data cache was reloaded from the local core's L2 cache with dispatch conflict due to a demand load.
PMC4	000004C042	PM_DATA_FROM_L3	The processor's data cache was reloaded from local core's L3 cache due to a demand load.

Table 5-6. Data Source Events (Sheet 3 of 3)

PMC	Event Code	Event Name	Event Description
PMC4	000004C044	PM_DATA_FROM_L3.1_ ECO_MOD	The processor's data cache was reloaded with modified data from another core's ECO L3 cache on the same chip due to a demand load.
PMC4	000004C046	PM_DATA_FROM_L2.1_ MOD	The processor's data cache was reloaded with modified data from another core's L2 cache on the same chip due to a demand load.
PMC4	000004C048	PM_DATA_FROM_DL2L3 _MOD	The processor's data cache was reloaded with modified data from another chip's L2 or L3 cache on a different node or group (distant) from this chip due to a demand load.
PMC4	000004C04A	PM_DATA_FROM_ OFF_CHIP_CACHE	The processor's data cache was reloaded either shared or modified data from another core's L2 or L3 cache on a different chip (remote or distant) due to a demand load.
PMC4	000004C04C	PM_DATA_FROM_ DMEM	The processor's data cache was reloaded from another chip's memory on the same node or group (distant) due to a demand load.
PMC4	000004C04E	PM_DATA_FROM_ L3MISS_MOD	The processor's data cache was reloaded from a location other than the local core's L3 cache due to a demand load.
PMC4	000004E04E	PM_DPTEG_FROM_ L3MISS	A PTE was loaded into the TLB from a location other than the local core's L3 cache due to a data-side request. When using Radix page translation, this count excludes PDE reloads. Only PTE reloads are included.
PMC4	000004C050	PM_DATA_SYS_PUMP_ MPRED_RTY	Final pump scope (system) ended up larger than the initial pump scope (chip/group) for a demand load.
PMC4	000004C052	PM_DATA_PUMP_ MPRED	Pump misprediction. Counts across all types of pumps for a demand load.
PMC4	00000400FE	PM_DATA_FROM_ MEMORY	The processor's data cache was reloaded from a memory location including the L4 cache from local remote or distant due to a demand load.
PMC3	000003E05E	PM_L3_CO_MEPF	L3 castouts in Mepf state for this thread.

5.7 Translation Events

As instruction and data addresses are processed by a core, they are typically translated from effective addresses to physical addresses. The one exception to this case is when translation is disabled, which is usually only used in some critical sections of low-level hypervisors or non-virtualized systems. On the POWER9 processor, three sets of structures exist as levels of the translation mechanism.

- ERAT
- TLB
- Page-walk cache

Each structure can translate some or all of the supported page sizes (4 KB, 64 KB, 2 MB, 16 MB, 1 GB, and 16 GB). Two separate modes of translation exist: radix and hash-page table. Both forms use the ERAT and TLB structures. Additionally, radix translation also employs the page-walk cache.

When translations cannot be resolved in these structures, they are passed down as memory accesses that might be resolved in lower levels of cache or memory. Further characterization of misses that progress down the memory hierarchy are detailed in the on-chip and off-chip memory counters. The overall impact of translation is captured in the CPI breakdown analysis.

Table 5-7 lists the translation events.

Table 5-7. Translation Events (Sheet 1 of 5)

PMC	Event Code	Event Name	Event Description
PMC1	0000015040	PM_IPTEG_FROM_L2_NO_CONFLICT	A PTE was loaded into the TLB from the local core's L2 cache without conflict due to a instruction-side request.
PMC1	0000015042	PM_IPTEG_FROM_L2	A PTE was loaded into the TLB from the local core's L2 cache due to an instruction-side request.
PMC1	0000015044	PM_IPTEG_FROM_L3_NO_CONFLICT	A PTE was loaded into the TLB from the local core's L3 cache without conflict due to an instruction-side request.
PMC1	0000015046	PM_IPTEG_FROM_L3.1_SHR	A PTE was loaded into the TLB with shared (S) data from another core's L3 cache on the same chip due to an instruction-side request.
PMC1	0000015048	PM_IPTEG_FROM_ON_CHIP_CACHE	A PTE was loaded into the TLB either shared or modified data from another core's L2 or L3 cache on the same chip due to an instruction-side request.
PMC1	000001504A	PM_IPTEG_FROM_RL2L3_SHR	A PTE was loaded into the TLB with shared (S) data from another chip's L2 or L3 cache on the same node or group (remote), as this chip due to an instruction-side request.
PMC1	000001504C	PM_IPTEG_FROM_LL4	A PTE was loaded into the TLB from the local chip's L4 cache due to an instruction-side request.
PMC1	000001504E	PM_IPTEG_FROM_L2MISS	A PTE was loaded into the TLB from a location other than the local core's L2 cache due to an instruction-side request.
PMC1	00000100F6	PM_IERAT_RELOAD	Number of I-ERAT reloads.
PMC2	0000025040	PM_IPTEG_FROM_L2_MEPF	A PTE was loaded into the TLB from local core's L2 hit without dispatch conflicts on Mepf state due to an instruction-side request.
PMC2	0000025042	PM_IPTEG_FROM_L3_MEPF	A PTE was loaded into the TLB from local core's L3 without dispatch conflicts hit on Mepf state due to an instruction-side request.
PMC2	0000025044	PM_IPTEG_FROM_L3.1_MOD	A PTE was loaded into the TLB with modified (M) data from another core's L3 cache on the same chip due to an instruction-side request.

Table 5-7. Translation Events (Sheet 2 of 5)

PMC	Event Code	Event Name	Event Description
PMC2	0000025046	PM_IPTEG_FROM_RL2L3_MOD	A PTE was loaded into the TLB with modified (M) data from another chip's L2 or L3 cache on the same node or group (remote), as this chip due to an instruction-side request.
PMC2	0000025048	PM_IPTEG_FROM_LMEM	A PTE was loaded into the TLB from the local chip's memory due to an instruction-side request.
PMC2	000002504A	PM_IPTEG_FROM_RL4	A PTE was loaded into the TLB from another chip's L4 cache on the same node or group (remote) due to an instruction-side request.
PMC2	000002504C	PM_IPTEG_FROM_MEMORY	A PTE was loaded into the TLB from a memory location including the L4 cache from local remote or distant due to an instruction-side request.
PMC2	0000020064	PM_IERAT_RELOAD_4K	I-ERAT reloaded (after a miss) for 4 KB pages.
PMC3	0000035042	PM_IPTEG_FROM_L3_DISP_CONFLICT	A PTE was loaded into the TLB from the local core's L3 cache with dispatch conflict due to an instruction-side request.
PMC3	0000035044	PM_IPTEG_FROM_L3.1_ECO_SHR	A PTE was loaded into the TLB with shared (S) data from another core's ECO L3 cache on the same chip due to an instruction-side request.
PMC3	0000035046	PM_IPTEG_FROM_L2.1_SHR	A PTE was loaded into the TLB with shared (S) data from another core's L2 cache on the same chip due to an instruction-side request.
PMC3	0000035048	PM_IPTEG_FROM_DL2L3_SHR	A PTE was loaded into the TLB with shared (S) data from another chip's L2 or L3 cache on a different node or group (distant) as this chip due to an instruction-side request.
PMC3	000003504A	PM_IPTEG_FROM_RMEM	A PTE was loaded into the TLB from another chip's memory on the same node or group (remote) due to an instruction-side request.
PMC3	000003504C	PM_IPTEG_FROM_DL4	A PTE was loaded into the TLB from another chip's L4 cache on a different node or group (distant) due to an instruction-side request.
PMC3	000003006A	PM_IERAT_RELOAD_64K	Instruction ERAT reloaded (miss) for a 64 KB page.
PMC4	0000040006	PM_ISLB_MISS	Number of instruction SLB misses for this thread.
PMC4	000004C054	PM_DERAT_MISS_16G_1G	Data ERAT miss (data TLB access) page size 16 GB (HPT mode) or 1 GB (radix mode).
PMC4	0000045042	PM_IPTEG_FROM_L3	A PTE was loaded into the TLB from local core's L3 cache due to an instruction-side request.
PMC4	0000045044	PM_IPTEG_FROM_L3.1_ECO_MOD	A PTE was loaded into the TLB with modified (M) data from another core's ECO L3 cache on the same chip due to an instruction-side request.
PMC4	0000045046	PM_IPTEG_FROM_L2.1_MOD	A PTE was loaded into the TLB with modified (M) data from another core's L2 cache on the same chip due to an instruction-side request.
PMC4	0000045048	PM_IPTEG_FROM_DL2L3_MOD	A PTE was loaded into the TLB with modified (M) data from another chip's L2 or L3 cache on a different node or group (distant), as this chip due to an instruction-side request.
PMC4	000004504A	PM_IPTEG_FROM_OFF_CHIP_CACHE	A PTE was loaded into the TLB either shared or modified data from another core's L2 or L3 cache on a different chip (remote or distant) due to an instruction-side request.
PMC4	000004504C	PM_IPTEG_FROM_DMEM	A PTE was loaded into the TLB from another chip's memory on the same node or group (distant) due to an instruction-side request.
PMC4	000004504E	PM_IPTEG_FROM_L3MISS	A PTE was loaded into the TLB from a location other than the local core's L3 cache due to an instruction-side request.
PMC4	000004006A	PM_IERAT_RELOAD_16M	I-ERAT reloaded (miss) for a 16 MB page.
PMC4	00000400FC	PM_ITLB_MISS	ITLB reloaded. Counts one per ITLB miss for HPT but multiple for radix depending on the number of levels traversed.

Table 5-7. Translation Events (Sheet 3 of 5)

PMC	Event Code	Event Name	Event Description
PMC1	000001E040	PM_DPTEG_FROM_L2_NO_CONFLICT	A PTE was loaded into the TLB from the local core's L2 cache without conflict due to a data-side request. When using radix page translation, this count excludes PDE reloads. Only PTE reloads are included.
PMC1	000001E042	PM_DPTEG_FROM_L2	A PTE was loaded into the TLB from the local core's L2 cache due to a data-side request. When using radix page translation, this count excludes PDE reloads. Only PTE reloads are included.
PMC1	000001E044	PM_DPTEG_FROM_L3_NO_CONFLICT	A PTE was loaded into the TLB from the local core's L3 cache without conflict due to a data side request. When using Radix page translation, this count excludes PDE reloads. Only PTE reloads are included.
PMC1	000001E046	PM_DPTEG_FROM_L3.1_SHR	A PTE was loaded into the TLB with shared (S) data from another core's L3 cache on the same chip due to a data-side request. When using Radix page translation, this count excludes PDE reloads. Only PTE reloads are included.
PMC1	000001E048	PM_DPTEG_FROM_ON_CHIP_CACHE	A PTE was loaded into the TLB either shared or modified data from another core's L2 or L3 cache on the same chip due to a data-side request. When using Radix page translation, this count excludes PDE reloads. Only PTE reloads are included.
PMC1	000001E04A	PM_DPTEG_FROM_RL2L3_SHR	A PTE was loaded into the TLB with shared (S) data from another chip's L2 or L3 cache on the same node or group (remote), as this chip due to a data side request. When using Radix page translation, this count excludes PDE reloads. Only PTE reloads are included.
PMC1	000001E04C	PM_DPTEG_FROM_LL4	A PTE was loaded into the TLB from the local chip's L4 cache due to a data-side request. When using Radix page translation, this count excludes PDE reloads. Only PTE reloads are included.
PMC1	000001E04E	PM_DPTEG_FROM_L2MISS	A PTE was loaded into the TLB from a location other than the local core's L2 cache due to a data side request. When using Radix page translation, this count excludes PDE reloads. Only PTE reloads are included.
PMC2	000002E040	PM_DPTEG_FROM_L2_MEPF	A PTE was loaded into the TLB from local core's L2 hit without dispatch conflicts on Mepf state. due to a data-side request. When using Radix page translation, this count excludes PDE reloads. Only PTE reloads are included.
PMC2	000002E042	PM_DPTEG_FROM_L3_MEPF	A PTE was loaded into the TLB from local core's L3 cache without dispatch conflicts hit on Mepf state due to a data-side request. When using Radix page translation, this count excludes PDE reloads. Only PTE reloads are included.
PMC2	000002E044	PM_DPTEG_FROM_L3.1_MOD	A PTE was loaded into the TLB with modified (M) data from another core's L3 cache on the same chip due to a data-side request. When using Radix page translation, this count excludes PDE reloads. Only PTE reloads are included.
PMC2	000002E046	PM_DPTEG_FROM_RL2L3_MOD	A PTE was loaded into the TLB with modified (M) data from another chip's L2 or L3 cache on the same node or group (remote), as this chip due to a data-side request. When using Radix page translation, this count excludes PDE reloads. Only PTE reloads are included.
PMC2	000002E048	PM_DPTEG_FROM_LMEM	A PTE was loaded into the TLB from the local chip's memory due to a data-side request. When using Radix page translation, this count excludes PDE reloads. Only PTE reloads are included.
PMC2	000002E04A	PM_DPTEG_FROM_RL4	A PTE was loaded into the TLB from another chip's L4 cache on the same node or group (remote) due to a data-side request. When using Radix page translation, this count excludes PDE reloads. Only PTE reloads are included.
PMC2	000002E04C	PM_DPTEG_FROM_MEMORY	A PTE was loaded into the TLB from a memory location including the L4 cache from local remote or distant due to a data-side request. When using Radix page translation, this count excludes PDE reloads. Only PTE reloads are included.
PMC3	000003E042	PM_DPTEG_FROM_L3_DISP_CONFLICT	A PTE was loaded into the TLB from the local core's L3 cache with dispatch conflict due to a data-side request. When using Radix page translation, this count excludes PDE reloads. Only PTE reloads are included.

Table 5-7. Translation Events (Sheet 4 of 5)

PMC	Event Code	Event Name	Event Description
PMC3	000003E044	PM_DPTEG_FROM_L3.1_ECO_SHR	A PTE was loaded into the TLB with shared (S) data from another core's ECO L3 cache on the same chip due to a data-side request. When using Radix page translation, this count excludes PDE reloads. Only PTE reloads are included.
PMC3	000003E046	PM_DPTEG_FROM_L2.1_SHR	A PTE was loaded into the TLB with shared (S) data from another core's L2 cache on the same chip due to a data-side request. When using Radix page translation, this count excludes PDE reloads. Only PTE reloads are included.
PMC3	000003E048	PM_DPTEG_FROM_DL2L3_SHR	A PTE was loaded into the TLB with shared (S) data from another chip's L2 or L3 cache on a different node or group (distant), as this chip due to a data-side request. When using Radix page translation, this count excludes PDE reloads. Only PTE reloads are included.
PMC3	000003E04A	PM_DPTEG_FROM_RMEM	A PTE was loaded into the TLB from another chip's memory on the same node or group (remote) due to a data-side request. When using Radix page translation, this count excludes PDE reloads. Only PTE reloads are included.
PMC3	000003E04C	PM_DPTEG_FROM_DL4	A PTE was loaded into the TLB from another chip's L4 cache on a different node or group (distant) due to a data-side request. When using Radix page translation, this count excludes PDE reloads. Only PTE reloads are included.
PMC4	000004E042	PM_DPTEG_FROM_L3	A PTE was loaded into the TLB from local core's L3 cache due to a data-side request. When using Radix page translation, this count excludes PDE reloads. Only PTE reloads are included.
PMC4	000004E044	PM_DPTEG_FROM_L3.1_ECO_MOD	A PTE was loaded into the TLB with modified (M) data from another core's ECO L3 cache on the same chip due to a data-side request. When using Radix page translation, this count excludes PDE reloads. Only PTE reloads are included.
PMC4	000004E046	PM_DPTEG_FROM_L2.1_MOD	A PTE was loaded into the TLB with modified (M) data from another core's L2 cache on the same chip due to a data-side request. When using Radix page translation, this count excludes PDE reloads. Only PTE reloads are included.
PMC4	000004E048	PM_DPTEG_FROM_DL2L3_MOD	A PTE was loaded into the TLB with modified (M) data from another chip's L2 or L3 cache on a different node or group (distant), as this chip due to a data-side request. When using Radix page translation, this count excludes PDE reloads. Only PTE reloads are included.
PMC4	000004E04A	PM_DPTEG_FROM_OFF_CHIP_CACHE	A PTE was loaded into the TLB with either shared or modified data from another core's L2 or L3 cache on a different chip (remote or distant) due to a data-side request. When using Radix page translation, this count excludes PDE reloads. Only PTE reloads are included.
PMC4	000004E04C	PM_DPTEG_FROM_DMEM	A PTE was loaded into the TLB from another chip's memory on the same node or group (distant) due to a data-side request. When using Radix page translation, this count excludes PDE reloads. Only PTE reloads are included.
PMC1	0000010016	PM_DSLB_MISS	SLB miss.
PMC1	0000010026	PM_TABLEWALK_CYC	Number of cycles that an instruction tablewalk is active.
PMC1	000001C056	PM_DERAT_MISS_4K	Data ERAT miss (data TLB access) page size 4 KB.
PMC1	000001F054	PM_TLB_HIT	Number of times the TLB had the data required by the instruction. Applies to both HPT and RPT.
PMC2	000002C054	PM_DERAT_MISS_64K	Data ERAT miss (data TLB access) page size 64 KB.
PMC2	000002C056	PM_DTLB_MISS_4K	Data TLB miss page size 4 KB.
PMC2	0000020066	PM_TLB_MISS	TLB miss (instruction and data).
PMC2	00000200F6	PM_LSU_DERAT_MISS	Data ERAT reloaded due to a data ERAT miss.
PMC3	000003C054	PM_DERAT_MISS_16M_2M	Data ERAT miss (data TLB access) for page size 16 MB (HPT mode) and 2 MB (radix mode).
Any PMC	000000D0A8	PM_DSLB_MISS	Counted when set to nest.

Table 5-7. Translation Events (Sheet 5 of 5)

PMC	Event Code	Event Name	Event Description
Any PMC	000000D8A8	PM_ISLB_MISS	Counted when set to nest.
Any PMC	000000E0BC	PM_LS0_PTE_ TABLEWALK_CYC	Indicates that a tablewalk on table0 is pending for the respective thread.
Any PMC	000000E8BC	PM_LS1_PTE_ TABLEWALK_CYC	Indicates that a tablewalk on table1 is pending for the respective thread.
Any PMC	000000F884	PM_TABLEWALK_CYC_ PREF	Tablewalk qualified for PTE prefetches.
Any PMC	000000F098	PM_XLATE_HPT_MODE	LSU reports every cycle that the thread is in HPT translation mode (not radix mode).
Any PMC	000000F898	PM_XLATE_RADIX_MOD E	LSU reports every cycle that the thread is in radix translation mode (not HPT mode).
Any PMC	000000F89C	PM_XLATE_MISS	Includes instruction, prefetch, and demand.

Table 5-8 lists the radix events.

Table 5-8. Radix Events (Sheet 1 of 2)

PMC	Event Code	Event Name	Event Description
PMC1	000001F058	PM_RADIX_PWC_L2_ PTE_FROM_L2	A page table entry (PTE) was reloaded to an L2 page-walk cache (PWC) from the core's L2 data cache. This implies that L3 and L4 PWC accesses were not necessary for this translation.
PMC1	000001F05A	PM_RADIX_PWC_L4_ PTE_FROM_L2	A PTE was reloaded to an L4 page-walk cache from the core's L2 data cache. This is the deepest level of PWC possible for a translation.
PMC1	000001F05C	PM_RADIX_PWC_L3_ PDE_FROM_L3	A page directory entry (PDE) was reloaded to an L3 page-walk cache from the core's L3 data cache.
PMC2	000002D026	PM_RADIX_PWC_L1_ PDE_FROM_L2	A page directory entry was reloaded to an L1 page-walk cache from the core's L2 data cache.
PMC2	000002D028	PM_RADIX_PWC_L2_ PDE_FROM_L2	A page directory entry was reloaded to an L2 page-walk cache from the core's L2 data cache.
PMC2	000002D02A	PM_RADIX_PWC_L3_ PDE_FROM_L2	A page directory entry was reloaded to an L3 page-walk cache from the core's L2 data cache.
PMC2	000002D02E	PM_RADIX_PWC_L3_ PTE_FROM_L2	A PTE was reloaded to an L3 page-walk cache from the core's L2 data cache. This implies that an L4 PWC access was not necessary for this translation.
PMC3	000003F054	PM_RADIX_PWC_L4_ PTE_FROM_L3MISS	A PTE was reloaded to an L4 page-walk cache from beyond the core's L3 data cache. This is the deepest level of PWC possible for a translation. The source could be local/remote/distant memory or another core's cache.
PMC3	000003F058	PM_RADIX_PWC_L1_ PDE_FROM_L3	A page directory entry was reloaded to an L1 page-walk cache from the core's L3 data cache.
PMC3	000003F05A	PM_RADIX_PWC_L2_ PDE_FROM_L3	A page directory entry was reloaded to an L2 page-walk cache from the core's L3 data cache.
PMC3	000003F05E	PM_RADIX_PWC_L3_ PTE_FROM_L3	A PTE was reloaded to an L3 page-walk cache from the core's L3 data cache. This implies that an L4 PWC access was not necessary for this translation.
PMC4	000004F054	PM_RADIX_PWC_MISS	A radix translation attempt missed in the TLB and all levels of page-walk cache.
PMC4	000004F056	PM_RADIX_PWC_L1_ PDE_FROM_L3MISS	A page directory entry was reloaded to an L1 page-walk cache from beyond the core's L3 data cache. The source could be local/remote/distant memory or another core's cache.

Table 5-8. Radix Events (Sheet 2 of 2)

PMC	Event Code	Event Name	Event Description
PMC4	000004F058	PM_RADIX_PWC_L2_ PTE_FROM_L3	A PTE was reloaded to an L2 page-walk cache from the core's L3 data cache. This implies that L3 and L4 PWC accesses were not necessary for this translation.
PMC4	000004F05A	PM_RADIX_PWC_L4_ PTE_FROM_L3	A PTE was reloaded to an L4 page-walk cache from the core's L3 data cache. This is the deepest level of PWC possible for a translation.
PMC4	000004F05C	PM_RADIX_PWC_L2_ PTE_FROM_L3MISS	A PTE was reloaded to an L2 page-walk cache from beyond the core's L3 data cache. This implies that L3 and L4 PWC accesses were not necessary for this translation. The source could be local/remote/distant memory or another core's cache.
PMC4	000004F05E	PM_RADIX_PWC_L3_ PTE_FROM_L3MISS	A PTE was reloaded to an L3 page-walk cache from beyond the core's L3 data cache. This implies that an L4 PWC access was not necessary for this translation. The source could be local/remote/distant memory or another core's cache.

5.8 L2 and L3 Events

Table 5-9 lists the L2 events and *Table 5-9* on page 58 lists the L3 events.

Table 5-9. L2 Events (Sheet 1 of 4)

PMC	Event Code	Event Name	Event Description
PMC1	0000016080	PM_L2_LD	All successful data-side load dispatches for this thread (L2 miss and L2 hits).
PMC1	0000016880	PM_L2_ST	All successful data-side store dispatches for this thread (L2 miss and L2 hits).
PMC2	0000026080	PM_L2_LD_MISS	All successful data-side load dispatches that were an L2 miss for this thread.
PMC2	0000026880	PM_L2_ST_MISS	All successful data-side store dispatches that were an L2 miss for this thread.
PMC3	0000036080	PM_L2_INST	All successful instruction-side instruction fetches (for example, i-dem, i-pref) dispatches for this thread.
PMC3	0000036880	PM_L2_INST_MISS	All successful instruction-side instruction-fetches (for example, i-dem, i-pref) dispatches for this thread that were an L2 miss.
PMC4	0000046080	PM_L2_DISP_ALL_ L2MISS	All successful data-side load/store or instruction-side instruction fetches dispatches for this thread that were an L2 miss.
PMC4	0000046880	PM_ISIDE_MRU_TOUCH	Instruction-side L2 MRU touch commands sent to the L2 cache for this thread.
PMC1	0000016082	PM_L2_CASTOUT_MOD	L2 castouts, modified (M, Mu, Me).
PMC1	0000016882	PM_L2_CASTOUT_SHR	L2 castouts, shared (Tx, Sx).
PMC2	0000026082	PM_L2_IC_INV	I-cache invalidates sent over the reload bus to the core.
PMC2	0000026882	PM_L2_DC_INV	D-cache invalidates sent over the reload bus to the core.
PMC3	0000036082	PM_L2_LD_DISP	All successful data-side load or instruction-side instruction-fetch dispatches for this thread.
PMC3	0000036882	PM_L2_LD_HIT	All successful data-side load or instruction-side instruction-fetch dispatches for this thread that were L2 hits.
PMC4	0000046082	PM_L2_ST_DISP	All successful data-side store dispatches for this thread.
PMC4	0000046882	PM_L2_ST_HIT	All successful data-side store dispatches for this thread that were L2 hits.
PMC1	0000016084	PM_L2_RCLD_DISP	All data-side load or instruction-side instruction-fetch dispatch attempts for this thread.

Table 5-9. L2 Events (Sheet 2 of 4)

PMC	Event Code	Event Name	Event Description
PMC1	0000016884	PM_L2_RCLD_DISP_FAIL_ADDR	All data-side load or instruction-side instruction-fetch dispatch attempts for this thread that failed due to an address collision conflicts with an L2 machine already working on this line (for example, load-hit-STQ or RC/CO/SN machines.
PMC2	0000026084	PM_L2_RCLD_DISP_FAIL_OTHER	All data-side load or instruction-side instruction-fetch dispatch attempts for this thread that failed due to reasons other than an address collision conflict with an L2 machines (for example, RC/CO machine not available).
PMC2	0000026884	PM_DSIDE_MRU_TOUCH	Data-side L2 MRU touch commands sent to the L2 cache.
PMC3	0000036084	PM_L2_RCST_DISP	All data-side store dispatch attempts for this thread.
PMC3	0000036884	PM_L2_RCST_DISP_FAIL_ADDR	All data-side store dispatch attempts for this thread that failed due to an address collision with RC/CO/SN/SQ.
PMC4	0000046084	PM_L2_RCST_DISP_FAIL_OTHER	All data-side store dispatch attempts for this thread that failed due to a reason other than address collision.
PMC1	0000016086	PM_L2_SN_M_WR_DONE	SNP dispatched for a store and was M (true M).
PMC1	0000016886	PM_CO_DISP_FAIL	CO dispatch failed due to all CO machines being busy.
PMC2	0000026086	PM_CO_TM_SC_FOOTPRINT	The L2 cache did a clean-if-dirty CO to the L3 cache (for example, created a store clean [SC] line in the L3 cache) or an L2 TM store-hit-dirty HPC line and L3 indicated an SC line formed in the L3 cache on the RDR bus.
PMC3	0000036086	PM_L2_RC_ST_DONE	RC did a store to a line that was Tx or Sx.
PMC3	0000036886	PM_L2_SN_SX_I_DONE	SNP dispatched and went from Sx to Ix.
PMC4	0000046086	PM_L2_SN_M_RD_DONE	SNP dispatched for a read and was in a modified state (true M).
PMC4	0000046886	PM_L2_SN_M_WR_DONE	SNP dispatched for a write and was M (true M). For DMA cache inject this will pulse if rty/push is required (will not pulse if cache inject is accepted).
PMC1	0000016088	PM_L2_LOC_GUESS_CORRECT	L2 guess local (LNS) and guess was correct (for example, data local).
PMC1	0000016888	PM_L2_LOC_GUESS_WRONG	L2 guess local (LNS) and guess was not correct (that is, the data is not on the chip).
PMC2	0000026088	PM_L2_GRP_GUESS_CORRECT	L2 guess group (GS or NNS) and guess was correct (data intra-group and not on-chip).
PMC2	0000026888	PM_L2_GRP_GUESS_WRONG	L2 guess group (GS or NNS) and guess was not correct (that is, the data is on-chip or beyond the group).
PMC3	0000036088	PM_L2_SYS_GUESS_CORRECT	L2 guess system (VGS or RNS) and guess was correct (that is, the data is beyond the group).
PMC3	0000036888	PM_L2_SYS_GUESS_WRONG	L2 guess system (VGS or RNS) and guess was not correct (that is, the data is not beyond the group).
PMC4	0000046088	PM_L2_CHIP_PUMP	RC requests that were local (also known as the chip) pump attempts.
PMC4	0000046888	PM_L2_GROUP_PUMP	RC requests that were on group (also known as the node) pump attempts.
PMC1	000001688A	PM_ISIDE_DISP	All instruction-side instruction-fetch dispatch attempts for this thread.
PMC2	000002608A	PM_ISIDE_DISP_FAIL_ADDR	All instruction-side instruction-fetch dispatch attempts for this thread that failed due to an address collision conflict with an L2 machine already working on this line (for example, load-hit-STQ or RC/CO/SN machines).
PMC2	000002688A	PM_ISIDE_DISP_FAIL_OTHER	All instruction-side instruction-fetch dispatch attempts for this thread that failed due to reasons other than an address collision conflict with an L2 machine (for example, no available RC/CO machines).
PMC3	000003608A	PM_L2_RTY_ST	RC retries on SMP interconnect for any store from the core (excludes DCBFs).

Table 5-9. L2 Events (Sheet 3 of 4)

PMC	Event Code	Event Name	Event Description
PMC3	000003688A	PM_L2_RTY_LD	RC retries on SMP interconnect for any load from the core (excludes DCBFs).
PMC4	000004688A	PM_L2_SYS_PUMP	RC requests that were system pump attempts.
PMC1	000001608C	PM_RC0_BUSY	RC machine 0 (mach0) busy. Used by the PMU to sample average RC lifetime (mach0 used as a sample point)
PMC1	000001688C	PM_RC_USAGE	Continuous 16-cycle (2:1) window where this signals rotates through sampling each RC machine.
PMC2	000002608C	PM_RC0_BUSY	RC mach0 busy. Used by the PMU to sample average RC lifetime (mach0 used as a sample point).
PMC2	000002688C	PM_CO_USAGE	Continuous 16-cycle (2:1) window where this signals rotates through sampling each CO machine.
PMC3	000003608C	PM_CO0_BUSY	CO mach0 busy. Used by the PMU to sample average CO lifetime (mach0 used as a sample point).
PMC3	000003688C	PM_SN_USAGE	Continuous 16-cycle (2:1) window where this signal rotates through sampling each SN machine.
PMC4	000004608C	PM_CO0_BUSY	CO mach0 busy. Used by the PMU to sample average CO lifetime (mach0 used as a sample point).
PMC1	000001608E	PM_ST_CAUSED_FAIL	Non-TM store caused any thread to fail.
PMC1	000001688E	PM_TM_LD_CAUSED_ FAIL	Non-TM load caused any thread to fail.
PMC2	000002608E	PM_TM_LD_CONF	TM load (fav or non-fav) ran into a conflict (failed).
PMC2	000002688E	PM_TM_FAV_CAUSED_ FAIL	TM load (fav) caused another thread to fail.
PMC3	000003608E	PM_TM_ST_CONF	TM store (fav or non-fav) ran into a conflict (failed).
PMC3	000003688E	PM_TM_ST_CAUSED_ FAIL	TM store (fav or non-fav) caused another thread to fail.
PMC4	000004608E	PM_TM_CAP_ OVERFLOW	TM footprint capacity overflow.
PMC1	0000016090	PM_SN0_BUSY	SN mach0 busy. Used by the PMU to sample average SN lifetime (mach0 used as a sample point).
PMC1	0000016890	PM_L1PF_L2MEMACC	Valid when the first beat of data comes in for an L1 prefetch where data came from memory.
PMC2	0000026090	PM_SN0_BUSY	SN mach0 busy. Used by the PMU to sample average SN lifetime (mach0 used as a sample point).
PMC2	0000026890	PM_ISIDE_L2MEMACC	Valid when the first beat of data comes in for an instruction-side fetch where data came from memory.
PMC1	0000016092	PM_L2_LD_MISS_128B	All successful data-side load dispatches that were an L2 miss (not Sx, Tx, Mx) for this thread and the RC calculated the request should be for 128 bytes (that is, M = 0).
PMC1	0000016892	PM_L2_ST_MISS_128B	All successful data-side store dispatches that were an L2 miss (not Sx, Tx, Mx) for this thread and the RC calculated the request should be for 128 bytes (that is, M = 0).
PMC2	0000026092	PM_L2_LD_MISS_64B	All successful data-side load dispatches that were an L2 miss (not Sx, Tx, Mx) for this thread and the RC calculated the request should be for 64 byte (that is, M = 1).
PMC2	0000026892	PM_L2_ST_MISS_64B	All successful data-side store dispatches that were an L2 miss (not Sx, Tx, Mx) for this thread and the RC calculated the request should be for 64 bytes (that is, M = 1).

Table 5-9. L2 Events (Sheet 4 of 4)

PMC	Event Code	Event Name	Event Description
PMC3	0000036092	PM_DSIDE_L2MEMACC	Valid when the first beat of data comes in for a data-side fetch, where data came exclusively from memory (excluding hpcread64 accesses), that is, total memory accesses by RCs.
PMC3	0000036892	PM_DSIDE_OTHER_64B_L2MEMACC	Valid when the first beat of data comes in for a data-side fetch, where data came exclusively from memory that was for hpc_read64, (RC had to fetch other 64 bytes of a line from MC); that is, the number of times RC had to go to memory to get the missing 64 bytes.
PMC1	000001609E	PM_L2_LD_DISP	All successful data-side load or instruction-side instruction-fetch dispatches for this thread.
PMC1	000001689E	PM_L2_ST_DISP	All successful data-side store dispatches for this thread.
PMC2	000002609E	PM_L2_LD_HIT	All successful data-side load or instruction-side instruction-fetch dispatches for this thread that were L2 hits.
PMC2	000002689E	PM_L2_ST_HIT	All successful data-side store dispatches for this thread that were L2 hits.
PMC3	000003609E	PM_L2_INST	All successful instruction-side instruction-fetches (for example, i-dem, i-pref) dispatches for this thread.
PMC3	000003689E	PM_L2_RTY_LD	RC retries on the SMP interconnect for any load from the core (excludes DCBFs).
PMC4	000004609E	PM_L2_INST_MISS	All successful instruction-side instruction-fetches (for example, i-dem, i-pref) dispatches for this thread that were an L2 miss.
PMC4	000004689E	PM_L2_RTY_ST	RC retries on SMP interconnect for any store from core (excludes DCBFs).

Table 5-10. L3 Events (Sheet 1 of 3)

PMC	Event Code	Event Name	Event Description
PMC1	00000160A0	PM_L3_PF_MISS_L3	L3 prefetch missed in the L3 cache.
PMC1	00000168A0	PM_L3_CO_MEPF	L3 CO of line in Mepf state (includes casthrough to memory). The Mepf state indicates that a line was brought in to satisfy an L3 prefetch request.
PMC2	00000260A0	PM_L3_CO_MEM	L3 CO to memory or of port 0 and 1 (lossy = might undercount if two cresp come in the same cycle).
PMC2	00000268A0	PM_L3_CO_L31	L3 CO to L3.1 or of port 0 and 1 (lossy = might undercount if two cresps come in the same cycle).
PMC3	00000360A0	PM_L3_PF_ON_CHIP_CACHE	L3 prefetch from on-chip cache.
PMC3	00000368A0	PM_L3_PF_OFF_CHIP_CACHE	L3 prefetch from off-chip cache.
PMC4	00000460A0	PM_L3_PF_ON_CHIP_MEM	L3 prefetch from on-chip memory.
PMC4	00000468A0	PM_L3_PF_OFF_CHIP_MEM	L3 prefetch from off-chip memory.
PMC2	00000260A2	PM_L3_CI_HIT	L3 castins hit (total count).
PMC2	00000268A2	PM_L3_CI_MISS	L3 castins miss (total count).
PMC3	00000360A2	PM_L3_L2_CO_HIT	L2 CO hits.
PMC3	00000368A2	PM_L3_L2_CO_MISS	L2 CO miss.
PMC4	00000460A2	PM_L3_LAT_CI_HIT	L3 lateral castins hit.
PMC4	00000468A2	PM_L3_LAT_CI_MISS	L3 lateral castins miss.
PMC1	00000160A4	PM_L3_HIT	L3 hits (L2 miss hitting L3, including data/instruction/xlate).
PMC1	00000168A4	PM_L3_MISS	L3 misses (L2 miss also missing L3, including data/instruction/xlate).
PMC2	00000260A4	PM_L3_LD_HIT	L3 hits for demand loads.
PMC2	00000268A4	PM_L3_LD_MISS	L3 misses for demand loads.
PMC3	00000360A4	PM_L3_CO_LCO	Total L3 COs occurred on LCO L3.1 (good cresp, might end up in memory on a retry).
PMC3	00000368A4	PM_L3_CINJ	L3 castin of cache inject.
PMC4	00000468A4	PM_L3_TRANS_PF	L3 transient prefetch received from L2.
PMC1	00000160A6	PM_TM_SC_CO	L3 castout of a line that was store-copy (original value of speculatively written line) in a transaction.
PMC1	00000168A6	PM_TM_CAM_OVERFLOW	L3 TM CAM is full when an L2 castout of the TM_SC line occurs. The line is pushed to memory.
PMC2	00000260A6	PM_NON_TM_RST_SC	Non-TM snoop hits the line in the L3 cache that is a TM_SC state and causes it to be invalidated.
PMC2	00000268A6	PM_TM_RST_SC	A TM snoop hits the line in the L3 cache that is TM_SC state and causes it to be invalidated.
PMC3	00000360A6	PM_SNP_TM_HIT_M	A TM snoop that is a store hits the line in the L3 cache in an M or Mu state (exclusive modified).
PMC3	00000368A6	PM_SNP_TM_HIT_T	A TM snoop that is a store hits the line in the L3 cache in a T, Tn, or Te state (shared modified).
PMC4	00000460A6	PM_RD_FORMING_SC	Does not occur.

Table 5-10. L3 Events (Sheet 2 of 3)

PMC	Event Code	Event Name	Event Description
PMC4	00000468A6	PM_RD_CLEARING_SC	A core TM load hits the line in the L3 cache in a TM_SC state and causes it to be invalidated.
PMC1	00000168A8	PM_L3_WI_USAGE	Lifetime, sample of write inject machine 0 valid.
PMC2	00000260A8	PM_L3_PF_HIT_L3	L3 prefetch hit in the L3 cache (abandoned).
PMC2	00000268A8	PM_RD_HIT_PF	Read machine hit the L3 prefetch machine.
PMC3	00000360A8	PM_L3_CO	L3 castout occurring (does not include casthrough or log writes).
PMC3	00000368A8	PM_SN_INVL	Any port snooper detects a store to a line in the Sx state and invalidates the line. Up to four can happen in a cycle but only one is counted.
PMC4	00000460A8	PM_SN_HIT	Any port snooper hits the L3 cache. Up to four can happen in a cycle but only one is counted.
PMC4	00000468A8	PM_SN_MISS	Any port snooper has an L3 miss or collision. Up to four can happen in a cycle but only one is counted.
PMC1	00000160AA	PM_L3_P0_LCO_NO_DATA	Dataless L3 LCO sent port 0.
PMC1	00000168AA	PM_L3_P1_LCO_NO_DATA	Dataless L3 LCO sent port 1.
PMC2	00000260AA	PM_L3_P0_LCO_DATA	LCO sent with data port 0.
PMC2	00000268AA	PM_L3_P1_LCO_DATA	LCO sent with data port 1.
PMC3	00000360AA	PM_L3_P0_CO_MEM	L3 CO to memory port 0 with or without data.
PMC3	00000368AA	PM_L3_P1_CO_MEM	L3 CO to memory port 1 with or without data.
PMC4	00000460AA	PM_L3_P0_CO_L31	L3 CO to L3.1 (LCO) port 0 with or without data.
PMC4	00000468AA	PM_L3_P1_CO_L31	L3 CO to L3.1 (LCO) port 1 with or without data.
PMC1	00000160AC	PM_L3_SN_USAGE	Rotating sample of 16 snoop valids.
PMC1	00000168AC	PM_L3_CI_USAGE	Rotating sample of 16 CI or CO actives.
PMC2	00000260AC	PM_L3_PF_USAGE	Rotating sample of 32 prefetch actives.
PMC2	00000268AC	PM_L3_RD_USAGE	Rotating sample of 16 read actives.
PMC3	00000360AC	PM_L3_SN0_BUSY	Lifetime, sample of snooper machine 0 valid.
PMC3	00000368AC	PM_L3_CO0_BUSY	Lifetime, sample of CO machine 0 valid
PMC4	00000460AC	PM_L3_SN0_BUSY	Lifetime, sample of snooper machine 0 valid.
PMC4	00000468AC	PM_L3_CO0_BUSY	Lifetime, sample of CO machine 0 valid.
PMC1	00000160AE	PM_L3_P0_PF_RTY	L3 prefetch received retry port 0, every retry counted.
PMC1	00000168AE	PM_L3_P1_PF_RTY	L3 PF received retry port 1, every retry counted.
PMC2	00000260AE	PM_L3_P2_PF_RTY	L3 PF received retry port 2, every retry counted.
PMC2	00000268AE	PM_L3_P3_PF_RTY	L3 PF received retry port 3, every retry counted.
PMC3	00000360AE	PM_L3_P0_CO_RTY	L3 CO received retry port 0 (memory only), every retry counted.
PMC3	00000368AE	PM_L3_P1_CO_RTY	L3 CO received retry port 1 (memory only), every retry counted.
PMC4	00000460AE	PM_L3_P2_CO_RTY	L3 CO received retry port 2 (memory only), every retry counted.
PMC4	00000468AE	PM_L3_P3_CO_RTY	L3 CO received retry port 3 (memory only), every retry counted.
PMC1	00000160B0	PM_L3_P0_NODE_PUMP	L3 prefetch sent with nodal scope port 0, counts even retried requests

Table 5-10. L3 Events (Sheet 3 of 3)

PMC	Event Code	Event Name	Event Description
PMC1	00000168B0	PM_L3_P1_NODE_PUMP	L3 prefetch sent with nodal scope port 1, counts even retried requests
PMC2	00000260B0	PM_L3_P0_GRP_PUMP	L3 prefetch sent with group scope port 0, counts even retried requests
PMC2	00000268B0	PM_L3_P1_GRP_PUMP	L3 prefetch sent with group scope port 1, counts even retried requests
PMC3	00000360B0	PM_L3_P0_SYS_PUMP	L3 prefetch sent with system scope port 0, counts even retried requests
PMC3	00000368B0	PM_L3_P1_SYS_PUMP	L3 prefetch sent with system scope port 1, counts even retried requests
PMC1	00000160B2	PM_L3_LOC_GUESS_ CORRECT	Prefetch scope predictor selected LNS and was correct.
PMC1	00000168B2	PM_L3_GRP_GUESS_ CORRECT	Prefetch scope predictor selected GS or NNS and was correct.
PMC2	00000260B2	PM_L3_SYS_GUESS_ CORRECT	Prefetch scope predictor selected VGS or RNS and was correct.
PMC2	00000268B2	PM_L3_LOC_GUESS_ WRONG	Prefetch scope predictor selected LNS, but was wrong.
PMC3	00000360B2	PM_L3_GRP_GUESS_ WRONG_LOW	Prefetch scope predictor selected GS or NNS, but was wrong because scope was LNS.
PMC3	00000368B2	PM_L3_GRP_GUESS_ WRONG_HIGH	Prefetch scope predictor selected GS or NNS, but was wrong because scope was VGS or RNS.
PMC4	00000460B2	PM_L3_SYS_GUESS_ WRONG	Prefetch scope predictor selected VGS or RNS, but was wrong.
PMC1	00000160B4	PM_L3_P0_LCO_RTY	L3 initiated LCO received retry on port 0 (can try four times).
PMC1	00000168B4	PM_L3_P1_LCO_RTY	L3 initiated LCO received retry on port 1 (can try four times).
PMC2	00000260B4	PM_L3_P2_LCO_RTY	L3 initiated LCO received retry on port 2 (can try four times).
PMC2	00000268B4	PM_L3_P3_LCO_RTY	L3 initiated LCO received retry on port 3 (can try four times).
PMC3	00000360B4	PM_L3_PF0_BUSY	Lifetime, sample of PF machine 0 valid.
PMC3	00000368B4	PM_L3_RD0_BUSY	Lifetime, sample of RD machine 0 valid.
PMC4	00000460B4	PM_L3_PF0_BUSY	Lifetime, sample of PF machine 0 valid.
PMC4	00000468B4	PM_L3_RD0_BUSY	Lifetime, sample of RD machine 0 valid.
PMC1	00000160B6	PM_L3_WI0_BUSY	Rotating sample of eight write inject (WI) valid.
PMC2	00000260B6	PM_L3_WI0_BUSY	Rotating sample of eight WI valid (duplicate).

5.9 CPI Stack Events

The simplest way to visualize processor performance is to consider that during every cycle the thread might complete one or more instructions or none at all. Identifying the cycles when no instructions complete or a suboptimal number of instructions complete is the basic step in tuning. Tuning can involve environmental changes, such as using more efficient page sizes for an application, or an actual program, or library changes that more efficiently flow through the processor cores.

The POWER9 processor is different from previous POWER processors in that it does not structurally organize instructions into groups for dispatch or completion tracking. Processors that use grouping typically must complete all of the instructions in the group simultaneously. This complicates analysis because a group's completion can be stalled for multiple reasons, which necessitated some sampling methodologies in the PMUs for those predecessor processors. The POWER9 processor is designed to track and complete instructions individually; therefore, the PMU can identify unique conditions when an individual instruction is not completed.

The POWER9 PMU provides functionality for a hardware-based CPI stack that can hierarchically account for completion stalls and front-end stalls on a per-thread basis. The POWER9 PMU CPI analysis uses a set of counters to count the cycles for an instruction to complete or that do not complete. On the cycles that instructions do not complete, the performance monitor identifies the reason. A CPI breakdown contains multiple levels. The first level simply identifies if an instruction completes in a cycle or provides a first-order categorization of why the next instruction to complete did not finish in that cycle. The lower levels of the analysis, when present, provide more granularity of the root cause.

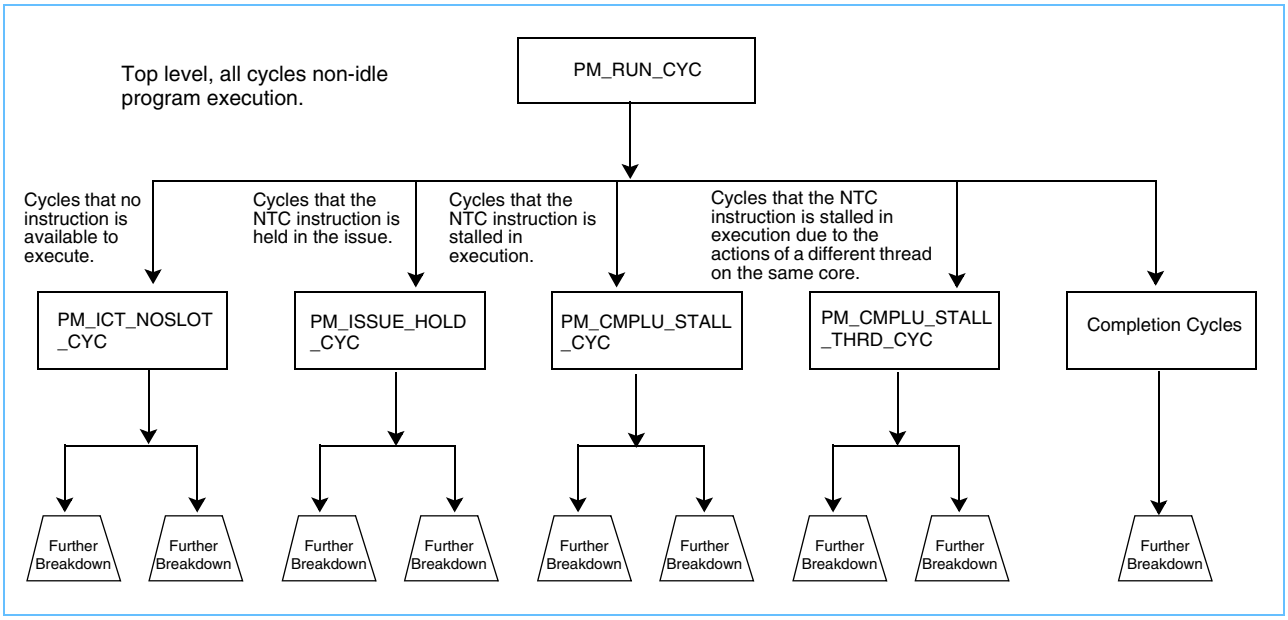
In reality, the CPI breakdown is best visualized as a tree. Each level of the tree brings further categorization of hardware performance. The sum of the cycles of a lower level of the tree should match the cycles of the higher-level component.

At the most basic level, the CPI stack (on a per-thread level) can be broken down into six main categories:

- ICT empty (no instructions have been dispatched for this thread).
- Issue hold (instructions are waiting to be issued to an execution unit).
- Pipeline stall (the instruction is being executed/waiting to satisfy a request).
- Thread blocked (exceptions and other blocks).
- Finish-to-completion (nominal completion pipeline cycles).
- Completion. Completion cycles are cycles when at least one instruction completed for that thread. The PMU can determines this using the completion interface from the ISU.

Figure 5-1 shows a simplified version of the categorization tree.

Figure 5-1. CPI Breakdown as a Tree



The ratio of any category to run cycles defines the fraction of cycles that the category represents. Tuning begins by identifying the most significant categories and drilling down to further levels for more specific reasons on why instructions are not completing. When specific categories require further correlation to program code, profiling can be employed on the related market events.

On the POWER9 processor, the CPI breakdown tree is a maximum of six levels deep. In some cases, the cycle accreditation of the components of a leaf is directly measured, and sometimes it is computed from other elements of the leaf.

Table 5-11 on page 67 shows all the events in CPI stack in the tree-format and *Table 5-12* on page 69 provides event codes and description of the events.

Table 5-11. CPI Stack Table (PM_RUN_CYC) (Sheet 1 of 2)

PM_RUN_CYC	PM_ICT_NOSLOT_CYC (PMC1)	PM_ICT_NOSLOT_IC_MISS	PM_ICT_NOSLOT_IC_L2		
			PM_ICT_NOSLOT_IC_L3		
			PM_ICT_NOSLOT_IC_L3MISS		
		PM_ICT_NOSLOT_BR_MPRED			
		PM_ICT_NOSLOT_BR_MPRED_ICMISS			
		PM_ICT_NOSLOT_DISP_HELD	PM_ICT_NOSLOT_DISP_HELD_HB_FULL		
			PM_ICT_NOSLOT_DISP_HELD_SYNC		
			PM_ICT_NOSLOT_DISP_HELD_TBEGIN		
			PM_ICT_NOSLOT_DISP_HELD_ISSQ		
			PM_ICT_NOSLOT_DISP_HELD_OTHER		
		Nothing to Dispatch (Other)			
	PM_ISSUE_HOLD	PM_NTC_ISSUE_HELD_DARQ_FULL			
		PM_NTC_ISSUE_HELD_ARB			
		PM_NTC_ISSUE_HELD_OTHER			
	PM_CMPLU_STALL (PMC4)	PM_CMPLU_STALL_BRU			
		PM_CMPLU_STALL_EXEC_UNIT	PM_CMPLU_STALL_SCALAR	PM_CMPLU_STALL_FXU (PMC2)	PM_CMPLU_STALL_FXLONG (PMC4)
					Stall due to single-cycle scalar fixed-point and CR instructions
				PM_CMPLU_STALL_DP (PMC1)	PM_CMPLU_STALL_DPLONG (PMC3)
					Stall due to a single-cycle scalar DP instruction
				PM_CMPLU_STALL_DFU (PMC2)	PM_CMPLU_STALL_DFLONG (PMC1)
					Stall due to a single-cycle scalar DFU instruction
				PM_CMPLU_STALL_PM (PMC3	
				PM_CMPLU_STALL_CRYPTO (PMC4)	
			PM_CMPLU_STALL_VECTOR	PM_CMPLU_STALL_VFX (PMC3)	PM_CMPLU_STALL_VFXLONG (PMC2)
					Stall due to a single-cycle vector fixed-point instruction
				PM_CMPLU_STALL_VDP (PMC4)	PM_CMPLU_STALL_VDPLONG (PMC3)
					Stall due to a single-cycle vector DP instruction
			Stall due to execution slices (other VSU/FXU/CR		
			PM_CMPLU_STALL_LSAQ	PM_CMPLU_STALL_LRQ_FULL	
				PM_CMPLU_STALL_SRQ_FULL	
				PM_CMPLU_STALL_LSAQ_ARB	
			PM_CMPLU_STALL_EMQ	PM_CMPLU_STALL_ERAT_MISS	
				PM_CMPLU_STALL_EMQ_FULL	
			PM_CMPLU_STALL_LRQ	PM_CMPLU_STALL_LMQ_FULL	
				PM_CMPLU_STALL_ST_FWD	
				PM_CMPLU_STALL_LHS	
				PM_CMPLU_STALL_LSU_MFSPR	
				PM_CMPLU_STALL_LARX	
				PM_CMPLU_STALL_LRQ_OTHER	

Table 5-11. CPI Stack Table (PM_RUN_CYC) (Sheet 2 of 2)

PM_RUN_CYC	PM_CMPLU_STALL (PMC4)		PM_CMPLU_STALL_ DCACHE_MISS	PM_CMPLU_STALL_ DMISS_L2L3	PM_CMPLU_STALL_DMISS_L2 L3_CONFLICT
					L2/L3 hit with no conflict
				PM_CMPLU_STALL_DMI SS_L3MISS	PM_CMPLU_STALL_DMISS_ L21_L31
					PM_CMPLU_STALL_DMISS_ LMEM
					PM_CMPLU_STALL_DMISS_ REMOTE
					Stall due to off-node memory cache
			PM_CMPLU_STALL_LOAD_FINISH		
			PM_CMPLU_STALL_ SRQ	PM_CMPLU_STALL_STORE_DATA	
				PM_CMPLU_STALL_LWSYNV	
				PM_CMPLU_STALL_HWSYNC	
				PM_CMPLU_STALL_EIEIO	
				PM_CMPLU_STALL_STCX	
				PM_CMPLU_STALL_SLB	
				PM_CMPLU_STALL_TEND	
				PM_CMPLU_STALL_PASTE	
				PM_CMPLU_STALL_TLBIE	
				PM_CMPLU_STALL_STORE_PIPE_ARB	
				PM_CMPLU_STALL_STORE_FIN_ARB	
			PM_CMPLU_STALL_STORE_FINISH		
			PM_CMPLU_STALL_LSU_FIN		
			Stall due to load/store (other)		
		PM_CMPLU_STALL_NTC_FLUSH			
		PM_CMPLU_STALL_NTC_DISP_FIN			
		Stall cycle other			
	PM_CMPLU_STALL_ THRD (PMC1)	PM_CMPLU_STALL_EXCEPTION			
		PM_CMPLU_STALL_ANY_SYNC			
		PM_CMPLU_STALL_SYNC_PMU_INT			
		PM_CMPLU_STALL_SPEC_FINISH			
		PM_CMPLU_STALL_FLUSH_ANY_THREAD			
		PM_CMPLU_STALL_LSU_FLUSH_NEXT			
		PM_CMPLU_STALL_NESTED_TBEGIN			
		PM_CMPLU_STALL_NESTED_TEND			
		PM_CMPLU_STALL_MTFPSCR			
		PM_CMPLU_STALL_OTHER_CMPL			
	PM_1PLUS_PPC_CMP				
	Other				

Table 5-12. CPI Stack Events (Sheet 1 of 4)

PMC	Event Code	Event Name	Event Description
PMC1	0000010004	PM_CMPLU_STALL_LRQ_OTHER	Finish stall due to miscellaneous LRQ reasons, lost arbitration to an LMQ slot, bank collisions, set prediction cleanup, set prediction multi-hit, and others.
PMC1	000001001C	PM_CMPLU_STALL_THRD	Completion stalled because the thread was blocked.
PMC1	000001002A	PM_CMPLU_STALL_LARX	Finish stall because the NTF instruction was a **larx** waiting to be satisfied.
PMC1	000001003A	PM_CMPLU_STALL_LSU_FIN	Finish stall because the NTF instruction was an LSU operation (other than a load or a store) with all of its dependencies met and just going through the LSU pipe to finish.
PMC1	000001003C	PM_CMPLU_STALL_DMISS_L2L3	Completion stall by D-cache miss that was resolved in the L2 or L3 cache.
PMC1	000001005A	PM_CMPLU_STALL_DFLONG	Finish stall because the NTF instruction was a multi-cycle instruction issued to the decimal floating-point execution pipe and waiting to finish. Includes decimal floating-point instructions and 128-bit binary floating-point instructions.
PMC1	000001005C	PM_CMPLU_STALL_DP	Finish stall because the NTF instruction was a scalar instruction issued to the double-precision execution pipe and waiting to finish. Includes binary floating-point instructions in 32- and 64-bit binary floating-point format. Not qualified multicycle.
PMC1	000001E050	PM_CMPLU_STALL_TEND	Finish stall because the NTF instruction was a **tend** instruction awaiting response from the L2 cache.
PMC1	000001E052	PM_CMPLU_STALL_SLB	Finish stall because the NTF instruction was waiting for an L2 response for an SLB.
PMC1	000001E054	PM_CMPLU_STALL	Nothing completed and ICT is not empty.
PMC1	000001E056	PM_CMPLU_STALL_FLUSH_ANY_THREAD	Cycles in which the NTC instruction is not allowed to complete because any of the four threads in the same core suffered a flush, which blocks completion.
PMC1	000001E05A	PM_CMPLU_STALL_ANY_SYNC	Cycles in which the NTC **sync** instruction (**isync**, **lwsync** or **hwsync**) is not allowed to complete.
PMC1	000001E05C	PM_CMPLU_STALL_NESTED_TBEGIN	Completion stall because the ISU is updating the TEXASR to keep track of the nested **tbegin**. This is a short delay and includes ROT.
PMC1	0000010064	PM_ICT_NOSLOT_DISP_HELD_TBEGIN	The NTC instruction is being held at dispatch because it is a **tbegin** instruction and there is an older **tbegin** in the pipeline that must complete before the younger **tbegin** can dispatch.
PMC1	00000100F8	PM_ICT_NOSLOT_CYC	Number of cycles the ICT has no itags assigned to this thread.
PMC2	000002C010	PM_CMPLU_STALL_LSU	Completion stalled by an LSU instruction.
PMC2	000002C012	PM_CMPLU_STALL_DCACHE_MISS	Finish stall because the NTF instruction was a load that missed the L1 cache and was waiting for the data to return from the nest.
PMC2	000002C014	PM_CMPLU_STALL_STORE_FINISH	Finish stall because the NTF instruction was a store with all its dependencies met, just waiting to go through the LSU pipe to finish
PMC2	000002C016	PM_CMPLU_STALL_PASTE	Finish stall because the NTF instruction was a paste waiting for a response from the L2 cache.
PMC2	000002C018	PM_CMPLU_STALL_DMISS_L21_L31	Completion stall by a D-cache miss that was resolved on chip (excluding local L2/L3 cache).
PMC2	000002C01A	PM_CMPLU_STALL_LHS	Finish stall because the NTF instruction was a load that hit on an older store and it was waiting for store data.
PMC2	000002C01C	PM_CMPLU_STALL_DMISS_REMOTE	Completion stall by a D-cache miss that was resolved from remote chip (cache or memory).

Table 5-12. CPI Stack Events (Sheet 2 of 4)

PMC	Event Code	Event Name	Event Description
PMC2	000002C01E	PM_CMPLU_STALL_SYNC_PMU_INT	Cycles in which the NTC instruction is waiting for a synchronous PMU interrupt.
PMC2	000002D012	PM_CMPLU_STALL_DFU	Finish stall because the NTF instruction was issued to the decimal floating-point execution pipe and waiting to finish. Includes decimal floating-point instructions and 128-bit binary floating-point instructions. Not qualified by multicycle.
PMC2	000002D014	PM_CMPLU_STALL_LRQ_FULL	Finish stall because the NTF instruction was a load that was held in the LSAQ because the LRQ was full.
PMC2	000002D016	PM_CMPLU_STALL_FXU	Finish stall due to a scalar fixed-point or CR instruction in the execution pipeline. These instructions get routed to the ALU, ALU2, and DIV pipes.
PMC2	000002D018	PM_CMPLU_STALL_EXEC_UNIT	Completion stall due to execution units (FXU/VSU/CRU).
PMC2	000002D01A	PM_ICT_NOSLOT_IC_MISS	ICT empty for this thread due to an I-cache miss.
PMC2	000002D01C	PM_CMPLU_STALL_STCX	Finish stall because the NTF instruction was a **stcx** waiting for a response from the L2 cache.
PMC2	000002D01E	PM_ICT_NOSLOT_DISP_HELD_ISSQ	ICT empty for this thread due to a dispatch hold on this thread due to an issue queue full, BRQ full, XVCF full, count cache, link, TAR full.
PMC2	000002E018	PM_CMPLU_STALL_VFXLONG	Completion stall due to a long latency vector fixed-point instruction (division, square root).
PMC2	000002E01A	PM_CMPLU_STALL_LSU_FLUSH_NEXT	Completion stall of one cycle because the LSU was requested to flush the next iop in the sequence. It takes one cycle for the ISU to process this request before the LSU instruction is allowed to complete.
PMC2	000002E01C	PM_CMPLU_STALL_TLBIE	Finish stall because the NTF instruction was a **tlbie** waiting for a response from the L2 cache.
PMC2	000002E01E	PM_CMPLU_STALL_NTC_FLUSH	Completion stall due to an NTC flush.
PMC3	0000030004	PM_CMPLU_STALL_EMQ_FULL	Finish stall because the NTF instruction suffered an ERAT miss and the EMQ was full.
PMC3	0000030006	PM_CMPLU_STALL_OTHER_CMPL	Instructions the core completed while this **tread** was stalled.
PMC3	000003000A	PM_CMPLU_STALL_PM	Finish stall because the NTF instruction was issued to the permute execution pipe and waiting to finish. Includes permute and decimal fixed-point instructions (128-bit BCD arithmetic) and a few 128-bit fixed-point add/subtract instructions with carry. Not qualified by vector or multicycle.
PMC3	0000030014	PM_CMPLU_STALL_STORE_FIN_ARB	Finish stall because the NTF instruction was a store waiting for a slot in the store finish pipe. This means the instruction is ready to finish but there are instructions ahead of it using the finish pipe.
PMC3	0000030016	PM_CMPLU_STALL_SRQ_FULL	Finish stall because the NTF instruction was a store that was held in the LSAQ because the SRQ was full.
PMC3	0000030018	PM_ICT_NOSLOT_DISP_HELD_HB_FULL	ICT empty for this thread due to dispatch holds because the history buffer was full. Can be GPR/VSR/VMR/FPR/CR/XVF; CR; XVF (XER/VSCR/FPSCR).
PMC3	0000030026	PM_CMPLU_STALL_STORE_DATA	Finish stall because the NTF instruction was a store waiting on data.
PMC3	0000030028	PM_CMPLU_STALL_SPEC_FINISH	Finish stall while waiting for the non-speculative finish of either a **stcx** waiting for its result or a load waiting for non-critical sectors of data and ECC.
PMC3	0000030038	PM_CMPLU_STALL_DMISS_LMEM	Completion stall due to cache miss that resolves in local memory.

Table 5-12. CPI Stack Events (Sheet 3 of 4)

PMC	Event Code	Event Name	Event Description
PMC3	000003003A	PM_CMPLU_STALL_EXCEPTION	Cycles that the NTC instruction is not allowed to complete because it was interrupted by any exception, which has to be serviced before the instruction can complete.
PMC3	000003003C	PM_CMPLU_STALL_NESTED_TEND	Completion stall because the ISU is updating the TEXASR to keep track of the nested **tend** and decrement the TEXASR nested level. This is a short delay.
PMC3	000003C05A	PM_CMPLU_STALL_VDPLONG	Finish stall because the NTF instruction was a scalar multi-cycle instruction issued to the double-precision execution pipe and waiting to finish. Includes binary floating-point instructions in 32- and 64-bit binary floating-point format. Qualified by NOT vector and multicycle.
PMC3	000003C05C	PM_CMPLU_STALL_VFXU	Finish stall due to a vector fixed-point instruction in the execution pipeline. These instructions get routed to the ALU, ALU2, and DIV pipes.
PMC3	0000034056	PM_CMPLU_STALL_LSU_MFSPR	Finish stall because the NTF instruction was an **mfspr** instruction targeting an LSU SPR and it was waiting for the register data to be returned.
PMC3	0000034058	PM_ICT_NOSLOT_BR_MPRED_ICMISS	ICT empty for this thread due to an I-cache miss and branch misprediction.
PMC3	000003405C	PM_CMPLU_STALL_DPLONG	Finish stall because the NTF instruction was a scalar multi-cycle instruction issued to the double-precision execution pipe and waiting to finish. Includes binary floating-point instructions in 32- and 64-bit binary floating-point format. Qualified by NOT vector AND multicycle.
PMC3	000003E052	PM_ICT_NOSLOT_IC_L3	ICT empty for this thread due to I-cache misses that were sourced from the local L3 cache.
PMC4	000004C010	PM_CMPLU_STALL_STORE_PIPE_ARB	Finish stall because the NTF instruction was a store waiting for the next relaunch opportunity after an internal reject. This means the instruction is ready to relaunch and tried once but lost arbitration.
PMC4	000004C012	PM_CMPLU_STALL_ERAT_MISS	Finish stall because the NTF instruction was a load or store that suffered a translation miss.
PMC4	000004C014	PM_CMPLU_STALL_LMQ_FULL	Finish stall because the NTF instruction was a load that missed in the L1 cache and the LMQ was unable to accept this load-miss request because it was full.
PMC4	000004C016	PM_CMPLU_STALL_DMISS_L2L3_CONFLICT	Completion stall due to a cache miss that resolves in the L2 or L3 cache with a conflict.
PMC4	000004C01A	PM_CMPLU_STALL_DMISS_L3MISS	Completion stall due to a cache miss resolving and missed the L3.
PMC4	000004C01C	PM_CMPLU_STALL_ST_FWD	Completion stall due to store forward.
PMC4	000004C01E	PM_CMPLU_STALL_CRYPTO	Finish stall because the NTF instruction was routed to the crypto execution pipeline and was waiting to finish.
PMC4	000004D014	PM_CMPLU_STALL_LOAD_FINISH	Finish stall because the NTF instruction was a load instruction with all of its dependencies satisfied and just going through the LSU pipe to finish.
PMC4	000004D016	PM_CMPLU_STALL_FXLONG	Completion stall due to a long latency scalar fixed-point instruction (division, square root).
PMC4	000004D018	PM_CMPLU_STALL_BRU	Completion stall due to a branch unit.
PMC4	000004D01A	PM_CMPLU_STALL_EIEIO	Finish stall because the NTF instruction is an EIEIO is waiting for a response from the L2 cache.
PMC4	000004D01C	PM_ICT_NOSLOT_DISP_HELD_SYNC	Dispatch held due to a synchronizing instruction at dispatch.
PMC4	000004D01E	PM_ICT_NOSLOT_BR_MPRED	ICT empty for this thread due to branch misprediction.

Table 5-12. CPI Stack Events (Sheet 4 of 4)

PMC	Event Code	Event Name	Event Description
PMC4	000004E010	PM_ICT_NOSLOT_IC_L3MISS	ICT empty for this thread due to an I-cache miss that was sourced from beyond the local L3 cache. The source can be local/remote/distant memory or another core's cache.
PMC4	000004E012	PM_CMPLU_STALL_MTFPSCR	Completion stall because the ISU is updating the register and notifying the effective address table (EAT).
PMC4	000004E016	PM_CMPLU_STALL_LSAQ_ARB	Finish stall because the NTF instruction was a load or store that was held in the LSAQ because an older instruction from the SRQ or LRQ won arbitration to the LSU pipe when this instruction tried to launch.
PMC4	000004E018	PM_CMPLU_STALL_NTC_DISP_FIN	Finish stall because the NTF instruction was one that must finish at dispatch.
PMC4	000004E01A	PM_ICT_NOSLOT_DISP_HELD	Cycles in which the NTC instruction is held at dispatch for any reason.
PMC4	000004405C	PM_CMPLU_STALL_VDP	Finish stall because the NTF instruction was a vector instruction issued to the double-precision execution pipe and waiting to finish. Includes binary floating-point instructions in 32- and 64-bit binary floating-point format. Not qualified multi-cycle. Qualified by vector.

5.10 Marked Events

Table 5-13 on page 72 lists the marked events.

Table 5-13. Marked Events (Sheet 1 of 8)

PMC	Event Code	Event Name	Event Description
PMC1	0000010132	PM_MRK_INST_ISSUED	Marked instruction issued.
PMC1	0000010138	PM_MRK_BR_2PATH	Marked branches that are not strongly biased
PMC1	000001013E	PM_MRK_LD_MISS_EXPOSED_CYC	Marked load exposed miss (use edge detect to count number).
PMC1	000001D140	PM_MRK_DATA_FROM_L3.1_MOD_CYC	Duration in cycles to reload with modified (M) data from another core's L3 cache on the same chip due to a marked load.
PMC1	000001D142	PM_MRK_DATA_FROM_L3.1_ECO_SHR_CYC	Duration in cycles to reload with shared (S) data from another core's ECO L3 cache on the same chip due to a marked load.
PMC1	000001D144	PM_MRK_DATA_FROM_L3_DISP_CONFLICT	The processor's data cache was reloaded from local core's L3 cache with dispatch conflict due to a marked load.
PMC1	000001D146	PM_MRK_DATA_FROM_MEMORY_CYC	Duration in cycles to reload from a memory location including the L4 cache from local remote or distant due to a marked load.
PMC1	000001D148	PM_MRK_DATA_FROM_RMEM	The processor's data cache was reloaded from another chip's memory on the same node or group (remote) due to a marked load.
PMC1	000001D14A	PM_MRK_DATA_FROM_RL2L3_MOD	The processor's data cache was reloaded with modified data from another chip's L2 or L3 cache on the same node or group (remote), from this chip due to a marked load.
PMC1	000001D14C	PM_MRK_DATA_FROM_LL4	The processor's data cache was reloaded from the local chip's L4 cache due to a marked load.
PMC1	000001D14E	PM_MRK_DATA_FROM_OFF_CHIP_CACHE_CYC	Duration in cycles to reload either shared or modified data from another core's L2 or L3 cache on a different chip (remote or distant) due to a marked load.

Table 5-13. Marked Events (Sheet 2 of 8)

PMC	Event Code	Event Name	Event Description
PMC1	000001F140	PM_MRK_DPTEG_FROM_L2_NO_CONFLICT	A PTE was loaded into the TLB from the local core's L2 cache without conflict due to a marked data-side request. When using Radix page translation, this count excludes PDE reloads. Only PTE reloads are included.
PMC1	000001F142	PM_MRK_DPTEG_FROM_L2	A PTE was loaded into the TLB from the local core's L2 cache due to a marked data-side request. When using Radix page translation, this count excludes PDE reloads. Only PTE reloads are included.
PMC1	000001F144	PM_MRK_DPTEG_FROM_L3_NO_CONFLICT	A PTE was loaded into the TLB from the local core's L3 cache without conflict due to a marked data-side request. When using Radix page translation, this count excludes PDE reloads. Only PTE reloads are included.
PMC1	000001F146	PM_MRK_DPTEG_FROM_L3.1_SHR	A PTE was loaded into the TLB with shared data from another core's L3 cache on the same chip due to a marked data-side request. When using Radix page translation, this count excludes PDE reloads. Only PTE reloads are included.
PMC1	000001F148	PM_MRK_DPTEG_FROM_ON_CHIP_CACHE	A PTE was loaded into the TLB either shared or modified data from another core's L2 or L3 cache on the same chip due to a marked data-side request. When using Radix page translation, this count excludes PDE reloads. Only PTE reloads are included.
PMC1	000001F14A	PM_MRK_DPTEG_FROM_RL2L3_SHR	A PTE was loaded into the TLB with shared data from another chip's L2 or L3 cache on the same node or group (remote) from this chip due to a marked data-side request. When using Radix page translation, this count excludes PDE reloads. Only PTE reloads are included.
PMC1	000001F14C	PM_MRK_DPTEG_FROM_LL4	A PTE was loaded into the TLB from the local chip's L4 cache due to a marked data-side request. When using Radix page translation, this count excludes PDE reloads. Only PTE reloads are included.
PMC1	000001F14E	PM_MRK_DPTEG_FROM_L2MISS	A PTE was loaded into the TLB from a location other than the local core's L2 cache due to a marked data-side request. When using Radix page translation, this count excludes PDE reloads. Only PTE reloads are included.
PMC1	000001015E	PM_MRK_FAB_RSP_RD_T_INTV	Sampled read got a T intervention.
PMC1	0000014156	PM_MRK_DATA_FROM_L2_CYC	Duration in cycles to reload from local core's L2 cache due to a marked load.
PMC1	0000014158	PM_MRK_DATA_FROM_L2_NO_CONFLICT_CYC	Duration in cycles to reload from local core's L2 cache without conflict due to a marked load.
PMC1	000001415A	PM_MRK_DATA_FROM_L2_DISP_CONFLICT_LDHITST_CYC	Duration in cycles to reload from local core's L2 cache with load-hit-store conflict due to a marked load.
PMC1	000001415C	PM_MRK_DATA_FROM_L3_MEPF_CYC	Duration in cycles to reload from local core's L3 cache without dispatch conflicts hit on Mepf state due to a marked load.
PMC1	000001415E	PM_MRK_DATA_FROM_L3MISS_CYC	Duration in cycles to reload from a location other than the local core's L3 cache due to a marked load.
PMC1	000001D150	PM_MRK_DATA_FROM_DL2L3_SHR	The processor's data cache was reloaded with shared data from another chip's L2 or L3 cache on a different node or group (distant) from this chip due to a marked load.
PMC1	000001D152	PM_MRK_DATA_FROM_DL4	The processor's data cache was reloaded from another chip's L4 cache on a different node or group (distant) due to a marked load.
PMC1	000001D154	PM_MRK_DATA_FROM_L2.1_SHR_CYC	Duration in cycles to reload with shared data from another core's L2 cache on the same chip due to a marked load.
PMC1	000001D156	PM_MRK_LD_MISS_L1_CYC	Marked load latency.
PMC1	000001D15C	PM_MRK_DTLB_MISS_1G	Marked data TLB reload (after a miss) page size 1 GB. Implies radix translation was used.

Table 5-13. Marked Events (Sheet 3 of 8)

PMC	Event Code	Event Name	Event Description
PMC1	000001D15E	PM_MRK_RUN_CYC	Run cycles in which a marked instruction is in the pipeline.
PMC1	0000015150	PM_SYNC_MRK_PROBE_NOP	Marked ProbeNops that can cause synchronous interrupts.
PMC1	0000015152	PM_SYNC_MRK_BR_LINK	Marked branch and link branch that can cause a synchronous interrupt.
PMC1	0000015154	PM_SYNC_MRK_L3MISS	Marked L3 cache misses that can throw a synchronous interrupt.
PMC1	0000015156	PM_SYNC_MRK_FX_DIVIDE	Marked fixed-point divide that can cause a synchronous interrupt.
PMC1	0000015158	PM_SYNC_MRK_L2HIT	Marked L2 hits that can throw a synchronous interrupt.
PMC1	000001515A	PM_SYNC_MRK_L2MISS	Marked L2 miss that can throw a synchronous interrupt.
PMC1	000001515C	PM_SYNC_MRK_BR_MPRED	Marked branch mispredict that can cause a synchronous interrupt.
PMC1	000001E15E	PM_MRK_L2_TM_REQ_ABORT	TM abort.
PMC1	000001F150	PM_MRK_ST_L2DISP_TO_CMPL_CYC	Cycles from L2 RC dispatch to L2 RC completion.
PMC1	000001F152	PM_MRK_FAB_RSP_BKILL_CYC	Cycles L2 RC took for a bkill.
PMC1	000001F15E	PM_MRK_PROBE_NOP_CMPL	Marked ProbeNops completed.
PMC1	000001016E	PM_MRK_BR_CMPL	Branch instruction completed.
PMC1	00000101E0	PM_MRK_INST_DISP	The thread has dispatched a randomly sampled marked instruction.
PMC1	00000101E2	PM_MRK_BR_TAKEN_CMPL	Marked branch taken completed.
PMC1	00000101E4	PM_MRK_L1_ICACHE_MISS	Sampled instruction suffered an I-cache miss.
PMC1	00000101EA	PM_MRK_L1_RELOAD_VALID	Marked demand reload.
PMC2	0000020114	PM_MRK_L2_RC_DISP	Marked instruction RC dispatched in the L2 cache.
PMC2	000002011C	PM_MRK_NTC_CYC	Cycles during which the marked instruction is next-to-complete (completion is held up because the marked instruction has not completed yet).
PMC2	000002C120	PM_MRK_DATA_FROM_L2_NO_CONFLICT	The processor's data cache was reloaded from the local core's L2 cache without a conflict due to a marked load.
PMC2	000002C122	PM_MRK_DATA_FROM_L3_DISP_CONFLICT_CYC	Duration in cycles to reload from local core's L3 cache with a dispatch conflict due to a marked load.
PMC2	000002C124	PM_MRK_DATA_FROM_L2_DISP_CONFLICT_OTHER	The processor's data cache was reloaded from local core's L2 cache with a dispatch conflict due to a marked load.
PMC2	000002C126	PM_MRK_DATA_FROM_L2	The processor's data cache was reloaded from local core's L2 cache due to a marked load.
PMC2	000002C128	PM_MRK_DATA_FROM_DL2L3_SHR_CYC	Duration in cycles to reload with shared data from another chip's L2 or L3 cache on a different node or group (distant) from this chip due to a marked load.
PMC2	000002C12A	PM_MRK_DATA_FROM_RMEM_CYC	Duration in cycles to reload from another chip's memory on the same node or group (remote) due to a marked load.

Table 5-13. Marked Events (Sheet 4 of 8)

PMC	Event Code	Event Name	Event Description
PMC2	000002C12C	PM_MRK_DATA_FROM_DL4_CYC	Duration in cycles to reload from another chip's L4 cache on a different node or group (distant) due to a marked load.
PMC2	000002C12E	PM_MRK_DATA_FROM_LL4_CYC	Duration in cycles to reload from the local chip's L4 cache due to a marked load.
PMC2	000002D120	PM_MRK_DATA_FROM_OFF_CHIP_CACHE	The processor's data cache was reloaded either shared or modified data from another core's L2 or L3 cache on a different chip (remote or distant) due to a marked load.
PMC2	0000020130	PM_MRK_INST_DECODED	An instruction was marked at decode time. Random instruction sampling (RIS) only.
PMC2	0000020132	PM_MRK_DFU_FIN	Decimal unit marked instruction finish.
PMC2	0000020134	PM_MRK_FXU_FIN	FXU marked instruction finish.
PMC2	0000020138	PM_MRK_ST_NEST	Marked store sent to nest.
PMC2	000002013A	PM_MRK_BRU_FIN	BRU marked instruction finish.
PMC2	000002D142	PM_MRK_DATA_FROM_L3_MEPF	The processor's data cache was reloaded from the local core's L3 cache without a dispatch conflict hit on an Mepf state due to a marked load.
PMC2	000002D144	PM_MRK_DATA_FROM_L3.1_MOD	The processor's data cache was reloaded with modified data from another core's L3 cache on the same chip due to a marked load.
PMC2	000002D148	PM_MRK_DATA_FROM_L2_DISP_CONFLICT_LDHITST	The processor's data cache was reloaded from local core's L2 cache with a load-hit-store conflict due to a marked load.
PMC2	000002D14A	PM_MRK_DATA_FROM_RL2L3_MOD_CYC	Duration in cycles to reload with modified data from another chip's L2 or L3 cache on the same node or group (remote) from this chip due to a marked load.
PMC2	000002D14C	PM_MRK_DATA_FROM_L3.1_ECO_SHR	The processor's data cache was reloaded with shared data from another core's ECO L3 cache on the same chip due to a marked load.
PMC2	000002D14E	PM_MRK_DATA_FROM_L2.1_SHR	The processor's data cache was reloaded with shared data from another core's L2 cache on the same chip due to a marked load.
PMC2	000002F140	PM_MRK_DPTEG_FROM_L2_MEPF	A PTE was loaded into the TLB from the local core's L2 hit without dispatch conflicts on an Mepf state due to a marked data-side request. When using Radix page translation, this count excludes PDE reloads. Only PTE reloads are included.
PMC2	000002F142	PM_MRK_DPTEG_FROM_L3_MEPF	A PTE was loaded into the TLB from the local core's L3 hit without dispatch conflicts on an Mepf state due to a marked data-side request. When using Radix page translation, this count excludes PDE reloads. Only PTE reloads are included.
PMC2	000002F144	PM_MRK_DPTEG_FROM_L3.1_MOD	A PTE was loaded into the TLB with modified data from another core's L3 cache on the same chip due to a marked data-side request. When using Radix page translation, this count excludes PDE reloads. Only PTE reloads are included.
PMC2	000002F146	PM_MRK_DPTEG_FROM_RL2L3_MOD	A PTE was loaded into the TLB with modified data from another chip's L2 or L3 cache on the same node or group (remote), from this chip due to a marked data-side request. When using Radix page translation, this count excludes PDE reloads. Only PTE reloads are included.
PMC2	000002F148	PM_MRK_DPTEG_FROM_LMEM	A PTE was loaded into the TLB from the local chip's memory due to a marked data-side request. When using Radix page translation, this count excludes PDE reloads. Only PTE reloads are included.
PMC2	000002F14A	PM_MRK_DPTEG_FROM_RL4	A PTE was loaded into the TLB from another chip's L4 cache on the same node or group (remote) due to a marked data-side request. When using Radix page translation, this count excludes PDE reloads. Only PTE reloads are included.

Table 5-13. Marked Events (Sheet 5 of 8)

PMC	Event Code	Event Name	Event Description
PMC2	000002F14C	PM_MRK_DPTEG_FROM_MEMORY	A PTE was loaded into the TLB from a memory location including L4 cache from local remote or distant due to a marked data-side request. When using Radix page translation, this count excludes PDE reloads. Only PTE reloads are included.
PMC2	000002015E	PM_MRK_FAB_RSP_RWITM_RTY	Sampled store did a rwitm and received a rty.
PMC2	0000024156	PM_MRK_STCX_FIN	Number of marked **stcx** instructions finished. This includes instructions in the speculative path of a branch that might be flushed.
PMC2	0000024158	PM_MRK_INST	An instruction was marked; includes both RIS at decode time and random event sampling (RES) at the time the configured event happens.
PMC2	000002D150	PM_MRK_DERAT_MISS_4K	Marked data ERAT miss (data TLB Access) page size 4 KB.
PMC2	000002D152	PM_MRK_DERAT_MISS_2M	Marked data ERAT miss (data TLB access) page size 2 MB. Implies radix translation.
PMC2	000002D154	PM_MRK_DERAT_MISS_64K	Marked data ERAT miss (data TLB Access) page size 64 KB.
PMC2	000002D156	PM_MRK_DTLB_MISS_4K	Marked data TLB Miss page size 4 KB.
PMC2	000002D15E	PM_MRK_DTLB_MISS_16G	Marked data TLB Miss page size 16 GB.
PMC2	000002F152	PM_MRK_FAB_RSP_DCLAIM_CYC	Cycles L2 RC took for a **dclaim**.
PMC2	00000201E0	PM_MRK_DATA_FROM_MEMORY	The processor's data cache was reloaded from a memory location including the L4 cache from local remote or distant due to a marked load.
PMC2	00000201E2	PM_MRK_LD_MISS_L1	Marked data L1 cache demand miss counted at execution time. Note that this count is per slice. If a load spans multiple slices, this event will increment multiple times for a single load.
PMC2	00000201E4	PM_MRK_DATA_FROM_L3MISS	The processor's data cache was reloaded from a location other than the local core's L3 cache due to a marked load.
PMC3	000003012A	PM_MRK_L2_RC_DONE	Marked RC done.
PMC3	000003012C	PM_MRK_ST_FWD	Marked store forwards.
PMC3	0000030130	PM_MRK_INST_FIN	Marked instruction finished.
PMC3	0000030132	PM_MRK_VSU_FIN	VSU marked instruction finish.
PMC3	0000030134	PM_MRK_ST_CMPL_INT	Marked store finished with intervention.
PMC3	000003013E	PM_MRK_STALL_CMPLU_CYC	Number of cycles the marked instruction is experiencing a stall while it is NTC.
PMC3	000003D140	PM_MRK_DATA_FROM_L2_DISP_CONFLICT_OTHER_CYC	Duration in cycles to reload from local core's L2 cache with a dispatch conflict due to a marked load.
PMC3	000003D142	PM_MRK_DATA_FROM_LMEM	The processor's data cache was reloaded from the local chip's memory due to a marked load.
PMC3	000003D144	PM_MRK_DATA_FROM_L2_MEPF_CYC	Duration in cycles to reload from local core's L2 hit without a dispatch conflict on an Mepf state due to a marked load.
PMC3	000003D146	PM_MRK_DATA_FROM_L3_NO_CONFLICT	The processor's data cache was reloaded from local core's L3 cache without conflict due to a marked load.
PMC3	000003D148	PM_MRK_DATA_FROM_L2.1_MOD_CYC	Duration in cycles to reload with modified data from another core's L2 cache on the same chip due to a marked load.

Table 5-13. Marked Events (Sheet 6 of 8)

PMC	Event Code	Event Name	Event Description
PMC3	000003D14C	PM_MRK_DATA_FROM_DMEM	The processor's data cache was reloaded from another chip's memory on the same node or group (distant) due to a marked load.
PMC3	000003D14E	PM_MRK_DATA_FROM_DL2L3_MOD	The processor's data cache was reloaded with modified data from another chip's L2 or L3 cache on a different node or group (distant) from this chip due to a marked load.
PMC3	000003F142	PM_MRK_DPTEG_FROM_L3_DISP_CONFLICT	A PTE was loaded into the TLB from local core's L3 cache with a dispatch conflict due to a marked data-side request. When using Radix page translation, this count excludes PDE reloads. Only PTE reloads are included.
PMC3	000003F144	PM_MRK_DPTEG_FROM_L3.1_ECO_SHR	A PTE was loaded into the TLB with shared data from another core's ECO L3 cache on the same chip due to a marked data-side request. When using Radix page translation, this count excludes PDE reloads. Only PTE reloads are included.
PMC3	000003F146	PM_MRK_DPTEG_FROM_L2.1_SHR	A PTE was loaded into the TLB with shared data from another core's L2 cache on the same chip due to a marked data-side request. When using Radix page translation, this count excludes PDE reloads. Only PTE reloads are included.
PMC3	000003F148	PM_MRK_DPTEG_FROM_DL2L3_SHR	A PTE was loaded into the TLB with shared data from another chip's L2 or L3 cache on a different node or group (distant) from this chip due to a marked data-side request. When using Radix page translation, this count excludes PDE reloads. Only PTE reloads are included.
PMC3	000003F14A	PM_MRK_DPTEG_FROM_RMEM	A PTE was loaded into the TLB from another chip's memory on the same node or group (remote) due to a marked data-side request. When using Radix page translation, this count excludes PDE reloads. Only PTE reloads are included.
PMC3	000003F14C	PM_MRK_DPTEG_FROM_DL4	A PTE was loaded into the TLB from another chip's L4 cache on a different node or group (distant) due to a marked data-side request. When using Radix page translation, this count excludes PDE reloads. Only PTE reloads are included.
PMC3	0000030154	PM_MRK_FAB_RSP_DCLAIM	Marked store must do a **dclaim**.
PMC3	000003015E	PM_MRK_FAB_RSP_CLAIM_RTY	Sampled store did a **rwitm** and got a rty.
PMC3	000003D152	PM_MRK_DERAT_MISS_1G	Marked data ERAT miss (data TLB access) page size 1 GB. Implies radix translation.
PMC3	000003D154	PM_MRK_DERAT_MISS_16M	Marked data ERAT miss (data TLB access) page size 16 MB.
PMC3	000003D156	PM_MRK_DTLB_MISS_64K	Marked data TLB Miss page size 64 KB.
PMC3	000003D15E	PM_MULT_MRK	Multiple marked instructions.
PMC3	0000035150	PM_MRK_DATA_FROM_RL2L3_SHR	The processor's data cache was reloaded with shared data from another chip's L2 or L3 cache on the same node or group (remote) from this chip due to a marked load.
PMC3	0000035152	PM_MRK_DATA_FROM_L2MISS_CYC	Duration in cycles to reload from a location other than the local core's L2 cache due to a marked load.
PMC3	0000035154	PM_MRK_DATA_FROM_L3_CYC	Duration in cycles to reload from local core's L3 cache due to a marked load.
PMC3	0000035156	PM_MRK_DATA_FROM_L3.1_SHR_CYC	Duration in cycles to reload with shared data from another core's L3 cache on the same chip due to a marked load.
PMC3	0000035158	PM_MRK_DATA_FROM_L3.1_ECO_MOD_CYC	Duration in cycles to reload with modified data from another core's ECO L3 cache on the same chip due to a marked load.
PMC3	000003515A	PM_MRK_DATA_FROM_ON_CHIP_CACHE_CYC	Duration in cycles to reload either shared or modified data from another core's L2 or L3 cache on the same chip due to a marked load.

Table 5-13. Marked Events (Sheet 7 of 8)

PMC	Event Code	Event Name	Event Description
PMC3	000003515C	PM_MRK_DATA_FROM_RL4	The processor's data cache was reloaded from another chip's L4 cache on the same node or group (remote) due to a marked load.
PMC3	000003515E	PM_MRK_BACK_BR_CMPL	Marked branch instruction completed with a target address less than current instruction address.
PMC3	000003E158	PM_MRK_STCX_FAIL	Marked **stcx** failed.
PMC3	000003F150	PM_MRK_ST_DRAIN_TO_L2DISP_CYC	Cycles to drain store from core to the L2 cache.
PMC3	0000030162	PM_MRK_LSU_DERAT_MISS	Marked data ERAT reload (miss) for any page size.
PMC3	00000301E2	PM_MRK_ST_CMPL	Marked store completed and sent to the Nest.
PMC3	00000301E4	PM_MRK_BR_MPRED_CMPL	Marked branch mispredicted.
PMC3	00000301E6	PM_MRK_DERAT_MISS	ERAT miss (TLB access), all page sizes.
PMC4	0000040116	PM_MRK_LARX_FIN	A **larx** instruction finished.
PMC4	0000040118	PM_MRK_DCACHE_RELOAD_INTV	Combined intervention event.
PMC4	000004E11E	PM_MRK_DATA_FROM_DMEM_CYC	Duration in cycles to reload from another chip's memory on the same node or group (distant) due to a marked load.
PMC4	000004C120	PM_MRK_DATA_FROM_L2_MEPF	The processor's data cache was reloaded from local core's L2 hit without a dispatch conflict on an Mepf state due to a marked load.
PMC4	000004C124	PM_MRK_DATA_FROM_L3_NO_CONFLICT_CYC	Duration in cycles to reload from local core's L3 cache without conflict due to a marked load.
PMC4	000004C12A	PM_MRK_DATA_FROM_RL2L3_SHR_CYC	Duration in cycles to reload with shared data from another chip's L2 or L3 cache on the same node or group (remote) from this chip due to a marked load.
PMC4	000004D124	PM_MRK_DATA_FROM_L3.1_SHR	The processor's data cache was reloaded with shared data from another core's L3 cache on the same chip due to a marked load.
PMC4	000004D128	PM_MRK_DATA_FROM_LMEM_CYC	Duration in cycles to reload from the local chip's memory due to a marked load.
PMC4	000004D12A	PM_MRK_DATA_FROM_RL4_CYC	Duration in cycles to reload from another chip's L4 cache on the same node or group (remote) due to a marked load.
PMC4	000004D12E	PM_MRK_DATA_FROM_DL2L3_MOD_CYC	Duration in cycles to reload with modified data from another chip's L2 or L3 cache on a different node or group (distant) from this chip due to a marked load.
PMC4	0000040132	PM_MRK_LSU_FIN	LSU marked instruction PPC finish.
PMC4	0000040134	PM_MRK_INST_TIMEO	Marked instruction finish timeout (instruction lost).
PMC4	000004013A	PM_MRK_IC_MISS	Marked instruction experienced I-cache miss.
PMC4	000004D140	PM_MRK_DATA_FROM_ON_CHIP_CACHE	The processor's data cache was reloaded either shared or modified data from another core's L2 or L3 cache on the same chip due to a marked load.
PMC4	000004D142	PM_MRK_DATA_FROM_L3	The processor's data cache was reloaded from local core's L3 cache due to a marked load.
PMC4	000004D144	PM_MRK_DATA_FROM_L3.1_ECO_MOD	The processor's data cache was reloaded with modified data from another core's ECO L3 cache on the same chip due to a marked load.
PMC4	000004D146	PM_MRK_DATA_FROM_L2.1_MOD	The processor's data cache was reloaded with modified data from another core's L2 cache on the same chip due to a marked load.

Table 5-13. Marked Events (Sheet 8 of 8)

PMC	Event Code	Event Name	Event Description
PMC4	000004F142	PM_MRK_DPTEG_FROM_L3	A PTE was loaded into the TLB from the local core's L3 cache due to a marked data-side request. When using Radix page translation, this count excludes PDE reloads. Only PTE reloads are included.
PMC4	000004F144	PM_MRK_DPTEG_FROM_L3.1_ECO_MOD	A PTE was loaded into the TLB with modified data from another core's ECO L3 cache on the same chip due to a marked data-side request. When using Radix page translation, this count excludes PDE reloads. Only PTE reloads are included.
PMC4	000004F146	PM_MRK_DPTEG_FROM_L2.1_MOD	A PTE was loaded into the TLB with modified data from another core's L2 cache on the same chip due to a marked data-side request. When using Radix page translation, this count excludes PDE reloads. Only PTE reloads are included.
PMC4	000004F148	PM_MRK_DPTEG_FROM_DL2L3_MOD	A PTE was loaded into the TLB with modified data from another chip's L2 or L3 cache on a different node or group (distant) from this chip due to a marked data-side request. When using Radix page translation, this count excludes PDE reloads. Only PTE reloads are included.
PMC4	000004F14A	PM_MRK_DPTEG_FROM_OFF_CHIP_CACHE	A PTE was loaded into the TLB either shared or modified data from another core's L2 or L3 cache on a different chip (remote or distant) due to a marked data-side request. When using Radix page translation, this count excludes PDE reloads. Only PTE reloads are included.
PMC4	000004F14C	PM_MRK_DPTEG_FROM_DMEM	A PTE was loaded into the TLB from another chip's memory on the same node or group (distant) due to a marked data-side request. When using Radix page translation, this count excludes PDE reloads. Only PTE reloads are included.
PMC4	000004F14E	PM_MRK_DPTEG_FROM_L3MISS	A PTE was loaded into the TLB from a location other than the local core's L3 cache due to a marked data-side request. When using Radix page translation, this count excludes PDE reloads. Only PTE reloads are included.
PMC4	0000040154	PM_MRK_FAB_RSP_BKILL	Marked store had to do a **bkill**.
PMC4	000004015E	PM_MRK_FAB_RSP_RD_RTY	Sampled L2 cache reads retry count.
PMC4	000004C15C	PM_MRK_DERAT_MISS_16G	Marked data ERAT miss (data TLB access) page size 16 GB.
PMC4	000004C15E	PM_MRK_DTLB_MISS_16M	Marked data TLB miss page size 16 MB.
PMC4	000004F150	PM_MRK_FAB_RSP_RWITM_CYC	Number of cycles an L2 RC took for a **rwitm**.
PMC4	00000401E0	PM_MRK_INST_CMPL	Marked instruction completed.
PMC4	00000401E4	PM_MRK_DTLB_MISS	Marked data TLB miss.
PMC4	00000401E6	PM_MRK_INST_FROM_L3MISS	Marked instruction was reloaded from a location beyond the local chiplet.
PMC4	00000401E8	PM_MRK_DATA_FROM_L2MISS	The processor's data cache was reloaded from a location other than the local core's L2 cache due to a marked load.

5.11 MMU Events

Table 5-14 lists the MMU events.

Table 5-14. MMU Events (Sheet 1 of 10)

PMC	Event Code	Event Name	Event Description
Special	0000008080	PM_RDXTLB_ANY_GST_ANY_HST_HITHIT	Any page-size guest TLB hit and any host page-size TLB hit on an initial TLB lookup.
Special	0000008082	PM_RDXTLB_ANY_GST_4K_HST_HITHIT	Any page-size guest TLB hit and 4 KB host page-size TLB hit on an initial TLB lookup.
Special	0000008084	PM_RDXTLB_ANY_GST_64K_HST_HITHIT	Any page-size guest TLB hit and 64 KB host page-size TLB hit on an initial TLB lookup.
Special	0000008086	PM_RDXTLB_ANY_GST_2M_HST_HITHIT	Any page-size guest TLB hit and 2 MB host page-size TLB hit on an initial TLB lookup.
Special	0000008088	PM_RDXTLB_ANY_GST_1G_HST_HITHIT	Any page-size guest TLB hit and 1 GB host page-size TLB hit on an initial TLB lookup.
Special	000000808A	PM_RDXTLB_ANY_GST_HITMISS	Any page-size guest TLB hit and a host TLB miss on an initial TLB lookup.
Special	000000888A	PM_RDXPWC_L1_GST_PWC_HIT	An L1 guest PWC hit.
Special	000000808C	PM_RDXTLB_EMQ_ACC	Any access to TLB from translations that are not table walks.
Special	000000888C	PM_RDXPWC_L2_GST_PWC_HIT	An L2 guest PWC hit.
Special	000000808E	PM_RDXTLB_MISS_ON_EMQ_ACC	Any miss to a TLB from translations that are not table walks.
Special	000000888E	PM_RDXPWC_L3_GST_PWC_HIT	An L3 guest PWC hit.
Special	0000008090	PM_RDXTLB_4K_GST_ANY_HST_HITHIT	4 KB page-size guest TLB hit and any host page-size TLB hit on an initial TLB lookup.
Special	0000008890	PM_RDXTLB_ANY_NSTD_ANY_HST_HIT	Any level of nested TLB lookup from a table walk that also had a host TLB hit.
Special	0000008092	PM_RDXTLB_4K_GST_4K_HST_HITHIT	A 4 KB page-size guest TLB hit and a 4 KB host page-size TLB hit on an initial TLB lookup.
Special	0000008892	PM_RDXTLB_ANY_NSTD_4K_HST_HIT	Any level of nested TLB lookup from a table walk that also had a host TLB hit on a 4 KB page size.
Special	0000008094	PM_RDXTLB_4K_GST_64K_HST_HITHIT	A 4 KB page-size guest TLB hit and a 64 KB host page-size TLB hit on an initial TLB lookup.
Special	0000008894	PM_RDXTLB_ANY_NSTD_64K_HST_HIT	Any level of nested TLB lookup from a table walk that also had a host TLB hit on a 64 KB page size.
Special	0000008096	PM_RDXTLB_4K_GST_2M_HST_HITHIT	A 4 KB page-size guest TLB hit and a 2 MB host page-size TLB hit on an initial TLB lookup.
Special	0000008896	PM_RDXTLB_ANY_NSTD_2M_HST_HIT	Any level of nested TLB lookup from a table walk that also had a host TLB hit on a 2 MB page size.
Special	0000008098	PM_RDXTLB_4K_GST_1G_HST_HITHIT	A 4 KB page-size guest TLB hit and a 1GB host page-size TLB hit on an initial TLB lookup.
Special	0000008898	PM_RDXTLB_ANY_NSTD_1G_HST_HIT	Any level of nested TLB lookup from a table walk that also had a host TLB hit on a 1 GB page size.
Special	000000809A	PM_RDXTLB_4K_GST_HITMISS	A 4 KB page-size guest TLB hit and a host TLB miss on an initial TLB lookup.

Table 5-14. MMU Events (Sheet 2 of 10)

PMC	Event Code	Event Name	Event Description
Special	000000889A	PM_RDXPWC_ANY_NSTD_L1_PWC_HIT	Any level of a nested TLB lookup from a table walk that also had an L1 PWC hit.
Special	000000809C	PM_RDXTLB_ANY_NSTD_ACC	Any level of nested TLB access from a table walk.
Special	000000889C	PM_RDXPWC_ANY_NSTD_L2_PWC_HIT	Any level of a nested TLB lookup from a table walk that also had an L2 PWC hit.
Special	000000809E	PM_RDXTLB_ANY_NSTD_MISS	Any level of a nested TLB access from a table walk that missed.
Special	000000889E	PM_RDXPWC_ANY_NSTD_L3_PWC_HIT	Any level of a nested TLB lookup from a table walk that also had an L3 PWC hit.
Special	00000080A0	PM_RDXTLB_64K_GST_ANY_HST_HITHIT	A 64 KB page-size guest TLB hit and any host page-size TLB hit on an initial TLB lookup.
Special	00000088A0	PM_RDXTLB_PRTE_NSTD_ANY_HST_HIT	Process table entry (PRTE) of a nested TLB lookup from a table walk that also had a host TLB hit.
Special	00000080A2	PM_RDXTLB_64K_GST_4K_HST_HITHIT	A 64 KB page-size guest TLB hit and a 4 KB host page-size TLB hit on an initial TLB lookup.
Special	00000088A2	PM_RDXTLB_PRTE_NSTD_4K_HST_HIT	PRTE of a nested TLB lookup from a table walk that also had a host TLB hit on a 4 KB page size.
Special	00000080A4	PM_RDXTLB_64K_GST_64K_HST_HITHIT	A 64 KB page-size guest TLB hit and a 64 KB host page-size TLB hit on an initial TLB lookup.
Special	00000088A4	PM_RDXTLB_PRTE_NSTD_64K_HST_HIT	PRTE of a nested TLB lookup from a table walk that also had a host TLB hit on a 64 KB page size.
Special	00000080A6	PM_RDXTLB_64K_GST_2M_HST_HITHIT	A 64 KB page-size guest TLB hit and a 2 MB host page-size TLB hit on an initial TLB lookup.
Special	00000088A6	PM_RDXTLB_PRTE_NSTD_2M_HST_HIT	PRTE of a nested TLB lookup from a table walk that also had a host TLB hit on a 2 MB page size.
Special	00000080A8	PM_RDXTLB_64K_GST_1G_HST_HITHIT	A 64 KB page-size guest TLB hit and a 1GB host page-size TLB hit on an initial TLB lookup.
Special	00000088A8	PM_RDXTLB_PRTE_NSTD_1G_HST_HIT	PRTE of a nested TLB lookup from a table walk that also had a host TLB hit on a 1 GB page size.
Special	00000080AA	PM_RDXTLB_64K_GST_HITMISS	A 64 KB page-size guest TLB hit and a host TLB miss on an initial TLB lookup.
Special	00000088AA	PM_RDXPWC_PRTE_NSTD_L1_PWC_HIT	PRTE of a nested TLB lookup from a table walk that also had an L1 PWC hit.
Special	00000080AC	PM_RDXTLB_PRTE_NSTD_ACC	PRTE of a nested TLB access from a table walk.
Special	00000088AC	PM_RDXPWC_PRTE_NSTD_L2_PWC_HIT	PRTE of a nested TLB lookup from a table walk that also had an L2 PWC hit.
Special	00000080AE	PM_RDXTLB_PRTE_NSTD_MISS	PRTE of a nested TLB access from a table walk that missed.
Special	00000088AE	PM_RDXPWC_PRTE_NSTD_L3_PWC_HIT	PRTE of a nested TLB lookup from a table walk that also had an L3 PWC hit.
Special	00000080B0	PM_RDXTLB_2M_GST_ANY_HST_HITHIT	A 2 MB page-size guest TLB hit and any host page-size TLB hit on an initial TLB lookup.
Special	00000088B0	PM_RDXTLB_LVL1_NSTD_ANY_HST_HIT	Level 1 of nested TLB lookup from table walk that also had a host TLB hit

Table 5-14. MMU Events (Sheet 3 of 10)

PMC	Event Code	Event Name	Event Description
Special	00000080B2	PM_RDXTLB_2M_GST_4K_HST_HITHIT	A 2 MB page-size guest TLB hit and a 4 KB host page-size TLB hit on an initial TLB lookup.
Special	00000088B2	PM_RDXTLB_LVL1_NSTD_4K_HST_HIT	Level 1 of a nested TLB lookup from a table walk that also had a host TLB hit on a 4 KB page size.
Special	00000080B4	PM_RDXTLB_2M_GST_64K_HST_HITHIT	A 2 MB page-size guest TLB hit and a 64 KB host page-size TLB hit on an initial TLB lookup.
Special	00000088B4	PM_RDXTLB_LVL1_NSTD_64K_HST_HIT	Level 1 of a nested TLB lookup from a table walk that also had a host TLB hit on a 64 KB page size.
Special	00000080B6	PM_RDXTLB_2M_GST_2M_HST_HITHIT	A 2 MB page-size guest TLB hit and a 2 MB host page-size TLB hit on an initial TLB lookup.
Special	00000088B6	PM_RDXTLB_LVL1_NSTD_2M_HST_HIT	Level 1 of a nested TLB lookup from a table walk that also had a host TLB hit on a 2 MB page size.
Special	00000080B8	PM_RDXTLB_2M_GST_1G_HST_HITHIT	A 2 MB page-size guest TLB hit and a 1 GB host page-size TLB hit on an initial TLB lookup.
Special	00000088B8	PM_RDXTLB_LVL1_NSTD_1G_HST_HIT	Level 1 of a nested TLB lookup from a table walk that also had a host TLB hit on a 1 GB page size.
Special	00000080BA	PM_RDXTLB_2M_GST_HITMISS	A 2 MB page-size guest TLB hit and a host TLB miss on an initial TLB lookup.
Special	00000088BA	PM_RDXPWC_LVL1_NSTD_L1_PWC_HIT	Level 1 of a nested TLB lookup from a table walk that also had an L1 PWC hit.
Special	00000080BC	PM_RDXTLB_LVL1_NSTD_ACC	Level 1 of a nested TLB access from a table walk.
Special	00000088BC	PM_RDXPWC_LVL1_NSTD_L2_PWC_HIT	Level 1 of a nested TLB lookup from a table walk that also had an L2 PWC hit.
Special	00000080BE	PM_RDXTLB_LVL1_NSTD_MISS	Level 1 of a nested TLB access from a table walk that missed.
Special	00000088BE	PM_RDXPWC_LVL1_NSTD_L3_PWC_HIT	Level 1 of a nested TLB lookup from a table walk that also had an L3 PWC hit.
Special	0000009080	PM_RDXTLB_1G_GST_ANY_HST_HITHIT	A 1 GB page-size guest TLB hit and any host page-size TLB hit on an initial TLB lookup.
Special	0000009880	PM_RDXTLB_LVL2_NSTD_ANY_HST_HIT	Level 2 of a nested TLB lookup from a table walk that also had a host TLB hit.
Special	0000009082	PM_RDXTLB_1G_GST_4K_HST_HITHIT	A 1 GB page-size guest TLB hit and a 4 KB host page-size TLB hit on an initial TLB lookup.
Special	0000009882	PM_RDXTLB_LVL2_NSTD_4K_HST_HIT	Level 2 of a nested TLB lookup from a table walk that also had a host TLB hit on a 4 KB page size.
Special	0000009084	PM_RDXTLB_1G_GST_64K_HST_HITHIT	A 1 GB page-size guest TLB hit and a 64 KB host page-size TLB hit on an initial TLB lookup.
Special	0000009884	PM_RDXTLB_LVL2_NSTD_64K_HST_HIT	Level 2 of a nested TLB lookup from a table walk that also had a host TLB hit on a 64 KB page size.
Special	0000009086	PM_RDXTLB_1G_GST_2M_HST_HITHIT	A 1 GB page-size guest TLB hit and a 2 MB host page-size TLB hit on an initial TLB lookup.
Special	0000009886	PM_RDXTLB_LVL2_NSTD_2M_HST_HIT	Level 2 of a nested TLB lookup from a table walk that also had a host TLB hit on a 2 MB page size.
Special	0000009088	PM_RDXTLB_1G_GST_1G_HST_HITHIT	A 1 GB page-size guest TLB hit and a 1 GB host page-size TLB hit on an initial TLB lookup.

Table 5-14. MMU Events (Sheet 4 of 10)

PMC	Event Code	Event Name	Event Description
Special	0000009888	PM_RDXTLB_LVL2_NSTD_1G_HST_HIT	Level 2 of a nested TLB lookup from a table walk that also had a host TLB hit on a 1 GB page size.
Special	000000908A	PM_RDXTLB_1G_GST_HITMISS	A 1 GB page-size guest TLB hit and a host TLB miss on an initial TLB lookup.
Special	000000988A	PM_RDXPWC_LVL2_NSTD_L1_PWC_HIT	Level 2 of a nested TLB lookup from a table walk that also had an L1 PWC hit.
Special	000000908C	PM_RDXTLB_LVL2_NSTD_ACC	Level 2 of a nested TLB access from table walk.
Special	000000988C	PM_RDXPWC_LVL2_NSTD_L2_PWC_HIT	Level 2 of a nested TLB lookup from table walk that also had an L2 PWC hit.
Special	000000908E	PM_RDXTLB_LVL2_NSTD_MISS	Level 2 of a nested TLB access from a table walk that missed.
Special	000000988E	PM_RDXPWC_LVL2_NSTD_L3_PWC_HIT	Level 2 of a nested TLB lookup from a table walk that also had an L3 PWC hit.
Special	0000009890	PM_RDXTLB_LVL3_NSTD_ANY_HST_HIT	Level 3 of a nested TLB lookup from a table walk that also had a host TLB hit.
Special	0000009892	PM_RDXTLB_LVL3_NSTD_4K_HST_HIT	Level 3 of a nested TLB lookup from a table walk that also had a host TLB hit on a 4 KB page size.
Special	0000009894	PM_RDXTLB_LVL3_NSTD_64K_HST_HIT	Level 3 of a nested TLB lookup from a table walk that also had a host TLB hit on a 64 KB page size.
Special	0000009896	PM_RDXTLB_LVL3_NSTD_2M_HST_HIT	Level 3 of a nested TLB lookup from a table walk that also had a host TLB hit on a 2 MB page size.
Special	0000009898	PM_RDXTLB_LVL3_NSTD_1G_HST_HIT	Level 3 of a nested TLB lookup from a table walk that also had a host TLB hit on a 1 GB page size.
Special	000000989A	PM_RDXPWC_LVL3_NSTD_L1_PWC_HIT	Level 3 of a nested TLB lookup from a table walk that also had an L1 PWC hit.
Special	000000909C	PM_RDXTLB_LVL3_NSTD_ACC	Level 3 of a nested TLB access from table walk.
Special	000000989C	PM_RDXPWC_LVL3_NSTD_L2_PWC_HIT	Level 3 of a nested TLB lookup from a table walk that also had an L2 PWC hit.
Special	000000909E	PM_RDXTLB_LVL3_NSTD_MISS	Level 3 of a nested TLB access from a table walk that missed.
Special	000000989E	PM_RDXPWC_LVL3_NSTD_L3_PWC_HIT	Level 3 of a nested TLB lookup from a table walk that also had an L3 PWC hit.
Special	00000098A0	PM_RDXTLB_LVL4_NSTD_ANY_HST_HIT	Level 4 of a nested TLB lookup from a table walk that also had a host TLB hit.
Special	00000098A2	PM_RDXTLB_LVL4_NSTD_4K_HST_HIT	Level 4 of a nested TLB lookup from a table walk that also had a host TLB hit on a 4 KB page size.
Special	00000098A4	PM_RDXTLB_LVL4_NSTD_64K_HST_HIT	Level 4 of a nested TLB lookup from a table walk that also had a host TLB hit on a 64 KB page size.
Special	00000098A6	PM_RDXTLB_LVL4_NSTD_2M_HST_HIT	Level 4 of a nested TLB lookup from a table walk that also had a host TLB hit on a 2 MB page size.
Special	00000098A8	PM_RDXTLB_LVL4_NSTD_1G_HST_HIT	Level 4 of a nested TLB lookup from a table walk that also had a host TLB hit on a 1 GB page size.
Special	00000098AA	PM_RDXPWC_LVL4_NSTD_L1_PWC_HIT	Level 4 of a nested TLB lookup from a table walk that also had an L1 PWC hit.

Table 5-14. MMU Events (Sheet 5 of 10)

PMC	Event Code	Event Name	Event Description
Special	00000090AC	PM_RDXTLB_LVL4_NSTD_ACC	Level 4 of a nested TLB access from table walk.
Special	00000098AC	PM_RDXPWC_LVL4_NSTD_L2_PWC_HIT	Level 4 of a nested TLB lookup from a table walk that also had an L2 PWC hit.
Special	00000090AE	PM_RDXTLB_LVL4_NSTD_MISS	Level 4 of a nested TLB access from a table walk that missed.
Special	00000098AE	PM_RDXPWC_LVL4_NSTD_L3_PWC_HIT	Level 4 of a nested TLB lookup from a table walk that also had an L3 PWC hit.
Special	00000098B0	PM_RDXTLB_FIN_NSTD_ANY_HST_HIT	Final translation of a nested TLB lookup from a table walk that also had a host TLB hit.
Special	00000098B2	PM_RDXTLB_FIN_NSTD_4K_HST_HIT	Final translation of a nested TLB lookup from a table walk that also had a host TLB hit on a 4 KB page size.
Special	00000098B4	PM_RDXTLB_FIN_NSTD_64K_HST_HIT	Final translation of a nested TLB lookup from a table walk that also had a host TLB hit on a 64 KB page size.
Special	00000098B6	PM_RDXTLB_FIN_NSTD_2M_HST_HIT	Final translation of a nested TLB lookup from a table walk that also had a host TLB hit on a 2 MB page size.
Special	00000098B8	PM_RDXTLB_FIN_NSTD_1G_HST_HIT	Final translation of a nested TLB lookup from a table walk that also had a host TLB hit on a 1 GB page size.
Special	00000098BA	PM_RDXPWC_FIN_NSTD_L1_PWC_HIT	Final translation of a nested TLB lookup from a table walk that also had an L1 PWC hit.
Special	00000090BC	PM_RDXTLB_FIN_NSTD_ACC	Final translation of a nested TLB access from a table walk.
Special	00000098BC	PM_RDXPWC_FIN_NSTD_L2_PWC_HIT	Final translation of a nested TLB lookup from a table walk that also had an L2 PWC hit.
Special	00000090BE	PM_RDXTLB_FIN_NSTD_MISS	Final translation of a nested TLB access from a table walk that missed.
Special	00000098BE	PM_RDXPWC_FIN_NSTD_L3_PWC_HIT	Final translation of a nested TLB lookup from a table walk that also had an L3 PWC hit.
Special	000000A080	PM_RDXWALK_TW0_ANY_GST_ANY_HST	Duration of a table walk for any guest level and any host level (not including outer-walk of guest memory accesses).
Special	000000A880	PM_RDXWALK_TW1_ANY_GST_ANY_HST	Duration of a table walk for any guest level and any host level (not including outer-walk of guest memory accesses).
Special	000000A082	PM_RDXWALK_TW0_ANY_GST_ANY_HST_WITH_GST_MEM_ACC	Duration of a table walk for any guest level and any host level (including outer-walk of guest memory accesses).
Special	000000A882	PM_RDXWALK_TW1_ANY_GST_ANY_HST_WITH_GST_MEM_ACC	Duration of a table walk for any guest level and any host level (including outer-walk of guest memory accesses).
Special	000000A084	PM_RDXWALK_TW0_ANY_GST_TLB_ACC_HST	Duration of a table walk for any guest level looking up a TLB (including outer-walk of guest memory accesses).
Special	000000A884	PM_RDXWALK_TW1_ANY_GST_TLB_ACC_HST	Duration of a table walk for any guest level looking up a TLB (including outer-walk of guest memory accesses).
Special	000000A086	PM_RDXWALK_TW0_ANY_GST_LVL1_ACC_HST	Duration of a table walk for any guest level, but limited to the duration only in level 1 host memory accesses and TLB/PWC reload.
Special	000000A886	PM_RDXWALK_TW1_ANY_GST_LVL1_ACC_HST	Duration of a table walk for any guest level, but limited to the duration only in level 1 host memory accesses and TLB/PWC reload.

Table 5-14. MMU Events (Sheet 6 of 10)

PMC	Event Code	Event Name	Event Description
Special	000000A088	PM_RDXWALK_TW0_ANY_GST_LVL2_ACC_HST	Duration of a table walk for any guest level, but limited to the duration only in level 2 host memory accesses and TLB/PWC reload.
Special	000000A888	PM_RDXWALK_TW1_ANY_GST_LVL2_ACC_HST	Duration of a table walk for any guest level, but limited to the duration only in level 2 host memory accesses and TLB/PWC reload.
Special	000000A08A	PM_RDXWALK_TW0_ANY_GST_LVL3_ACC_HST	Duration of a table walk for any guest level, but limited to the duration only in level 3 host memory accesses and TLB/PWC reload.
Special	000000A88A	PM_RDXWALK_TW1_ANY_GST_LVL3_ACC_HST	Duration of a table walk for any guest level, but limited to the duration only in level 3 host memory accesses and TLB/PWC reload.
Special	000000A08C	PM_RDXWALK_TW0_ANY_GST_LVL4_ACC_HST	Duration of a table walk for any guest level, but limited to the duration only in level 4 host memory accesses and TLB/PWC reload.
Special	000000A88C	PM_RDXWALK_TW1_ANY_GST_LVL4_ACC_HST	Duration of a table walk for any guest level, but limited to the duration only in level 4 host memory accesses and TLB/PWC reload.
Special	000000A08E	PM_RDXWALK_TW0_ANY_GST_MEM_ACC	Duration of a table walk for any guest level, but limited to the duration only of memory access and TLB/PWC reload of the guest entry.
Special	000000A88E	PM_RDXWALK_TW1_ANY_GST_MEM_ACC	Duration of a table walk for any guest level, but limited to the duration only of memory access and TLB/PWC reload of the guest entry.
Special	000000A090	PM_RDXWALK_TW0_PRTE_GST_ANY_HST	Duration of a table walk for guest PRTE translation and any host level (not including outer-walk of guest memory accesses).
Special	000000A890	PM_RDXWALK_TW1_PRTE_GST_ANY_HST	Duration of a table walk for guest PRTE translation and any host level (not including outer-walk of guest memory accesses).
Special	000000A092	PM_RDXWALK_TW0_PRTE_GST_ANY_HST_WITH_GST_MEM_ACC	Duration of a table walk for guest PRTE translation and any host level (including outer-walk of guest memory accesses).
Special	000000A892	PM_RDXWALK_TW1_PRTE_GST_ANY_HST_WITH_GST_MEM_ACC	Duration of a table walk for guest PRTE translation and any host level (including outer-walk of guest memory accesses).
Special	000000A094	PM_RDXWALK_TW0_PRTE_GST_TLB_ACC_HST	Duration of a table walk for guest PRTE translation looking up a TLB (including outer-walk of guest memory accesses).
Special	000000A894	PM_RDXWALK_TW1_PRTE_GST_TLB_ACC_HST	Duration of a table walk for guest PRTE translation looking up a TLB (including outer-walk of guest memory accesses).
Special	000000A096	PM_RDXWALK_TW0_PRTE_GST_LVL1_ACC_HST	Duration of a table walk for guest PRTE translation, but limited to the duration only in level-1 host memory accesses and TLB/PWC reload.
Special	000000A896	PM_RDXWALK_TW1_PRTE_GST_LVL1_ACC_HST	Duration of a table walk for guest PRTE translation, but limited to the duration only in level-1 host memory accesses and TLB/PWC reload.
Special	000000A098	PM_RDXWALK_TW0_PRTE_GST_LVL2_ACC_HST	Duration of a table walk for guest PRTE translation, but limited to the duration only in level-2 host memory accesses and TLB/PWC reload.
Special	000000A898	PM_RDXWALK_TW1_PRTE_GST_LVL2_ACC_HST	Duration of a table walk for guest PRTE translation, but limited to the duration only in level-2 host memory accesses and TLB/PWC reload.
Special	000000A09A	PM_RDXWALK_TW0_PRTE_GST_LVL3_ACC_HST	Duration of a table walk for guest PRTE translation, but limited to the duration only in level-3 host memory accesses and TLB/PWC reload.
Special	000000A89A	PM_RDXWALK_TW1_PRTE_GST_LVL3_ACC_HST	Duration of a table walk for guest PRTE translation, but limited to the duration only in level-3 host memory accesses and TLB/PWC reload.
Special	000000A09C	PM_RDXWALK_TW0_PRTE_GST_LVL4_ACC_HST	Duration of a table walk for guest PRTE translation, but limited to the duration only in level-4 host memory accesses and TLB/PWC reload.
Special	000000A89C	PM_RDXWALK_TW1_PRTE_GST_LVL4_ACC_HST	Duration of a table walk for guest PRTE translation, but limited to the duration only in level-4 host memory accesses and TLB/PWC reload.

Table 5-14. MMU Events (Sheet 7 of 10)

PMC	Event Code	Event Name	Event Description
Special	000000A09E	PM_RDXWALK_TW0_PRTE_GST_MEM_ACC	Duration of a table walk for guest PRTE translation, but limited to the duration only of memory access and TLB/PWC reload of the guest entry.
Special	000000A89E	PM_RDXWALK_TW1_PRTE_GST_MEM_ACC	Duration of a table walk for guest PRTE translation, but limited to the duration only of memory access and TLB/PWC reload of the guest entry
Special	000000A0A0	PM_RDXWALK_TW0_LVL1_GST_ANY_HST	Duration of a table walk for level-1 guest translation and any host level (not including outer-walk of guest memory accesses).
Special	000000A8A0	PM_RDXWALK_TW1_LVL1_GST_ANY_HST	Duration of a table walk for level-1 guest translation and any host level (not including outer-walk of guest memory accesses).
Special	000000A0A2	PM_RDXWALK_TW0_LVL1_GST_ANY_HST_WITH_GST_MEM_ACC	Duration of a table walk for level-1 guest translation and any host level (including outer-walk of guest memory accesses).
Special	000000A8A2	PM_RDXWALK_TW1_LVL1_GST_ANY_HST_WITH_GST_MEM_ACC	Duration of a table walk for level-1 guest translation and any host level (including outer-walk of guest memory accesses).
Special	000000A0A4	PM_RDXWALK_TW0_LVL1_GST_TLB_ACC_HST	Duration of a table walk for level-1 guest translation looking up a TLB (including outer-walk of guest memory accesses).
Special	000000A8A4	PM_RDXWALK_TW1_LVL1_GST_TLB_ACC_HST	Duration of a table walk for level-1 guest translation looking up a TLB (including outer-walk of guest memory accesses).
Special	000000A0A6	PM_RDXWALK_TW0_LVL1_GST_LVL1_ACC_HST	Duration of a table walk for level-1 guest translation, but limited to the duration only in level-1 host memory accesses and TLB/PWC reload.
Special	000000A8A6	PM_RDXWALK_TW1_LVL1_GST_LVL1_ACC_HST	Duration of a table walk for level-1 guest translation, but limited to the duration only in level-1 host memory accesses and TLB/PWC reload.
Special	000000A0A8	PM_RDXWALK_TW0_LVL1_GST_LVL2_ACC_HST	Duration of a table walk for level-1 guest translation, but limited to the duration only in level-2 host memory accesses and TLB/PWC reload.
Special	000000A8A8	PM_RDXWALK_TW1_LVL1_GST_LVL2_ACC_HST	Duration of a table walk for level-1 guest translation, but limited to the duration only in level-2 host memory accesses and TLB/PWC reload.
Special	000000A0AA	PM_RDXWALK_TW0_LVL1_GST_LVL3_ACC_HST	Duration of a table walk for level-1 guest translation, but limited to the duration only in level-3 host memory accesses and TLB/PWC reload.
Special	000000A8AA	PM_RDXWALK_TW1_LVL1_GST_LVL3_ACC_HST	Duration of a table walk for level-1 guest translation, but limited to the duration only in level-3 host memory accesses and TLB/PWC reload.
Special	000000A0AC	PM_RDXWALK_TW0_LVL1_GST_LVL4_ACC_HST	Duration of a table walk for level-1 guest translation, but limited to the duration only in level-4 host memory accesses and TLB/PWC reload.
Special	000000A8AC	PM_RDXWALK_TW1_LVL1_GST_LVL4_ACC_HST	Duration of a table walk for level-1 guest translation, but limited to the duration only in level-4 host memory accesses and TLB/PWC reload.
Special	000000A0AE	PM_RDXWALK_TW0_LVL1_GST_MEM_ACC	Duration of a table walk for level-1 guest translation, but limited to the duration only of memory access and TLB/PWC reload of the guest entry.
Special	000000A8AE	PM_RDXWALK_TW1_LVL1_GST_MEM_ACC	Duration of a table walk for level-1 guest translation, but limited to the duration only of memory access and TLB/PWC reload of the guest entry.
Special	000000A0B0	PM_RDXWALK_TW0_LVL2_GST_ANY_HST	Duration of a table walk for level-2 guest translation and any host level (not including outer-walk of guest memory accesses).
Special	000000A8B0	PM_RDXWALK_TW1_LVL2_GST_ANY_HST	Duration of a table walk for level-2 guest translation and any host level (not including outer-walk of guest memory accesses).
Special	000000A0B2	PM_RDXWALK_TW0_LVL2_GST_ANY_HST_WITH_GST_MEM_ACC	Duration of a table walk for level-2 guest translation and any host level (including outer-walk of guest memory accesses).
Special	000000A8B2	PM_RDXWALK_TW1_LVL2_GST_ANY_HST_WITH_GST_MEM_ACC	Duration of a table walk for level-2 guest translation and any host level (including outer-walk of guest memory accesses).

Table 5-14. MMU Events (Sheet 8 of 10)

PMC	Event Code	Event Name	Event Description
Special	000000A0B4	PM_RDXWALK_TW0_LVL2_GST_TLB_ACC_HST	Duration of a table walk for level-2 guest translation looking up a TLB (including outer-walk of guest memory accesses).
Special	000000A8B4	PM_RDXWALK_TW1_LVL2_GST_TLB_ACC_HST	Duration of a table walk for level-2 guest translation looking up a TLB (including outer-walk of guest memory accesses).
Special	000000A0B6	PM_RDXWALK_TW0_LVL2_GST_LVL1_ACC_HST	Duration of a table walk for level-2 guest translation, but limited to the duration only in level-1 host memory accesses and TLB/PWC reload.
Special	000000A8B6	PM_RDXWALK_TW1_LVL2_GST_LVL1_ACC_HST	Duration of a table walk for level-2 guest translation, but limited to the duration only in level-1 host memory accesses and TLB/PWC reload.
Special	000000A0B8	PM_RDXWALK_TW0_LVL2_GST_LVL2_ACC_HST	Duration of a table walk for level-2 guest translation, but limited to the duration only in level-2 host memory accesses and TLB/PWC reload.
Special	000000A8B8	PM_RDXWALK_TW1_LVL2_GST_LVL2_ACC_HST	Duration of a table walk for level-2 guest translation, but limited to the duration only in level-2 host memory accesses and TLB/PWC reload.
Special	000000A0BA	PM_RDXWALK_TW0_LVL2_GST_LVL3_ACC_HST	Duration of a table walk for level-2 guest translation, but limited to the duration only in level-3 host memory accesses and TLB/PWC reload.
Special	000000A8BA	PM_RDXWALK_TW1_LVL2_GST_LVL3_ACC_HST	Duration of a table walk for level-2 guest translation, but limited to the duration only in level-3 host memory accesses and TLB/PWC reload.
Special	000000A0BC	PM_RDXWALK_TW0_LVL2_GST_LVL4_ACC_HST	Duration of a table walk for level-2 guest translation, but limited to the duration only in level-4 host memory accesses and TLB/PWC reload.
Special	000000A8BC	PM_RDXWALK_TW1_LVL2_GST_LVL4_ACC_HST	Duration of a table walk for level-2 guest translation, but limited to the duration only in level-4 host memory accesses and TLB/PWC reload.
Special	000000A0BE	PM_RDXWALK_TW0_LVL2_GST_MEM_ACC	Duration of a table walk for level-2 guest translation, but limited to the duration only of memory access and TLB/PWC reload of the guest entry.
Special	000000A8BE	PM_RDXWALK_TW1_LVL2_GST_MEM_ACC	Duration of a table walk for level-2 guest translation, but limited to the duration only of memory access and TLB/PWC reload of the guest entry.
Special	000000B080	PM_RDXWALK_TW0_LVL3_GST_ANY_HST	Duration of a table walk for level-3 guest translation and any host level (not including outer-walk of guest memory accesses).
Special	000000B880	PM_RDXWALK_TW1_LVL3_GST_ANY_HST	Duration of a table walk for level-3 guest translation and any host level (not including outer-walk of guest memory accesses).
Special	000000B082	PM_RDXWALK_TW0_LVL3_GST_ANY_HST_WITH_GST_MEM_ACC	Duration of a table walk for level-3 guest translation and any host level (including outer-walk of guest memory accesses).
Special	000000B882	PM_RDXWALK_TW1_LVL3_GST_ANY_HST_WITH_GST_MEM_ACC	Duration of a table walk for level-3 guest translation and any host level (including outer-walk of guest memory accesses).
Special	000000B084	PM_RDXWALK_TW0_LVL3_GST_TLB_ACC_HST	Duration of a table walk for level-3 guest translation looking up TLB (including outer-walk of guest memory accesses).
Special	000000B884	PM_RDXWALK_TW1_LVL3_GST_TLB_ACC_HST	Duration of a table walk for level-3 guest translation looking up TLB (including outer-walk of guest memory accesses).
Special	000000B086	PM_RDXWALK_TW0_LVL3_GST_LVL1_ACC_HST	Duration of a table walk for level-3 guest translation, but limited to the duration only in level-1 host memory accesses and TLB/PWC reload.
Special	000000B886	PM_RDXWALK_TW1_LVL3_GST_LVL1_ACC_HST	Duration of a table walk for level-3 guest translation, but limited to the duration only in level-1 host memory accesses and TLB/PWC reload.
Special	000000B088	PM_RDXWALK_TW0_LVL3_GST_LVL2_ACC_HST	Duration of a table walk for level-3 guest translation, but limited to the duration only in level-2 host memory accesses and TLB/PWC reload.
Special	000000B888	PM_RDXWALK_TW1_LVL3_GST_LVL2_ACC_HST	Duration of a table walk for level-3 guest translation, but limited to the duration only in level-2 host memory accesses and TLB/PWC reload.

Table 5-14. MMU Events (Sheet 9 of 10)

PMC	Event Code	Event Name	Event Description
Special	000000B08A	PM_RDXWALK_TW0_LVL3_GST_LVL3_ACC_HST	Duration of a table walk for level-3 guest translation, but limited to the duration only in level-3 host memory accesses and TLB/PWC reload.
Special	000000B88A	PM_RDXWALK_TW1_LVL3_GST_LVL3_ACC_HST	Duration of a table walk for level-3 guest translation, but limited to the duration only in level-3 host memory accesses and TLB/PWC reload.
Special	000000B08C	PM_RDXWALK_TW0_LVL3_GST_LVL4_ACC_HST	Duration of a table walk for level-3 guest translation, but limited to the duration only in level-4 host memory accesses and TLB/PWC reload.
Special	000000B88C	PM_RDXWALK_TW1_LVL3_GST_LVL4_ACC_HST	Duration of a table walk for level-3 guest translation, but limited to the duration only in level-4 host memory accesses and TLB/PWC reload.
Special	000000B08E	PM_RDXWALK_TW0_LVL3_GST_MEM_ACC	Duration of a table walk for level-3 guest translation, but limited to the duration only of memory access and TLB/PWC reload of the guest entry.
Special	000000B88E	PM_RDXWALK_TW1_LVL3_GST_MEM_ACC	Duration of a table walk for level-3 guest translation, but limited to the duration only of memory access and TLB/PWC reload of the guest entry.
Special	000000B090	PM_RDXWALK_TW0_LVL4_GST_ANY_HST	Duration of a table walk for level-4 guest translation and any host level (not including outer-walk of guest memory accesses).
Special	000000B890	PM_RDXWALK_TW1_LVL4_GST_ANY_HST	Duration of a table walk for level-4 guest translation and any host level (not including outer-walk of guest memory accesses).
Special	000000B092	PM_RDXWALK_TW0_LVL4_GST_ANY_HST_WITH_GST_MEM_ACC	Duration of a table walk for level-4 guest translation and any host level (including outer-walk of guest memory accesses).
Special	000000B892	PM_RDXWALK_TW1_LVL4_GST_ANY_HST_WITH_GST_MEM_ACC	Duration of a table walk for level-4 guest translation and any host level (including outer-walk of guest memory accesses).
Special	000000B094	PM_RDXWALK_TW0_LVL4_GST_TLB_ACC_HST	Duration of a table walk for level-4 guest translation looking up TLB (including outer-walk of guest memory accesses).
Special	000000B894	PM_RDXWALK_TW1_LVL4_GST_TLB_ACC_HST	Duration of a table walk for level-4 guest translation looking up TLB (including outer-walk of guest memory accesses).
Special	000000B096	PM_RDXWALK_TW0_LVL4_GST_LVL1_ACC_HST	Duration of a table walk for level-4 guest translation, but limited to the duration only in level-1 host memory accesses and TLB/PWC reload.
Special	000000B896	PM_RDXWALK_TW1_LVL4_GST_LVL1_ACC_HST	Duration of a table walk for level-4 guest translation, but limited to the duration only in level-1 host memory accesses and TLB/PWC reload.
Special	000000B098	PM_RDXWALK_TW0_LVL4_GST_LVL2_ACC_HST	Duration of a table walk for level-4 guest translation, but limited to the duration only in level-2 host memory accesses and TLB/PWC reload.
Special	000000B898	PM_RDXWALK_TW1_LVL4_GST_LVL2_ACC_HST	Duration of a table walk for level-4 guest translation, but limited to the duration only in level-2 host memory accesses and TLB/PWC reload.
Special	000000B09A	PM_RDXWALK_TW0_LVL4_GST_LVL3_ACC_HST	Duration of a table walk for level-4 guest translation, but limited to the duration only in level-3 host memory accesses and TLB/PWC reload.
Special	000000B89A	PM_RDXWALK_TW1_LVL4_GST_LVL3_ACC_HST	Duration of a table walk for level-4 guest translation, but limited to the duration only in level-3 host memory accesses and TLB/PWC reload.
Special	000000B09C	PM_RDXWALK_TW0_LVL4_GST_LVL4_ACC_HST	Duration of a table walk for level-4 guest translation, but limited to the duration only in level-4 host memory accesses and TLB/PWC reload.
Special	000000B89C	PM_RDXWALK_TW1_LVL4_GST_LVL4_ACC_HST	Duration of a table walk for level-4 guest translation, but limited to the duration only in level-4 host memory accesses and TLB/PWC reload.
Special	000000B09E	PM_RDXWALK_TW0_LVL4_GST_MEM_ACC	Duration of a table walk for level-4 guest translation, but limited to the duration only of memory access and TLB/PWC reload of the guest entry.
Special	000000B89E	PM_RDXWALK_TW1_LVL4_GST_MEM_ACC	Duration of a table walk for level-4 guest translation, but limited to the duration only of memory access and TLB/PWC reload of the guest entry.

Table 5-14. MMU Events (Sheet 10 of 10)

PMC	Event Code	Event Name	Event Description
Special	000000B0A0	PM_RDXWALK_TW0_FIN_GST_ANY_HST	Duration of a table walk for level-4 guest translation and any host level (not including outer-walk of guest memory accesses).
Special	000000B8A0	PM_RDXWALK_TW1_FIN_GST_ANY_HST	Duration of a table walk for final guest translation and any host level (not including outer-walk of guest memory accesses).
Special	000000B0A2	PM_RDXWALK_TW0_FIN_GST_ANY_HST_WITH_GST_MEM_ACC	Duration of a table walk for final guest translation and any host level (including outer-walk of guest memory accesses).
Special	000000B8A2	PM_RDXWALK_TW1_FIN_GST_ANY_HST_WITH_GST_MEM_ACC	Duration of a table walk for final guest translation and any host level (including outer-walk of guest memory accesses).
Special	000000B0A4	PM_RDXWALK_TW0_FIN_GST_TLB_ACC_HST	Duration of a table walk for final guest translation looking up TLB (including outer-walk of guest memory accesses).
Special	000000B8A4	PM_RDXWALK_TW1_FIN_GST_TLB_ACC_HST	Duration of a table walk for final guest translation looking up TLB (including outer-walk of guest memory accesses).
Special	000000B0A6	PM_RDXWALK_TW0_FIN_GST_LVL1_ACC_HST	Duration of a table walk for final guest translation, but limited to the duration only in level-1 host memory accesses and TLB/PWC reload.
Special	000000B8A6	PM_RDXWALK_TW1_FIN_GST_LVL1_ACC_HST	Duration of a table walk for final guest translation, but limited to the duration only in level-1 host memory accesses and TLB/PWC reload.
Special	000000B0A8	PM_RDXWALK_TW0_FIN_GST_LVL2_ACC_HST	Duration of a table walk for final guest translation, but limited to the duration only in level-2 host memory accesses and TLB/PWC reload.
Special	000000B8A8	PM_RDXWALK_TW1_FIN_GST_LVL2_ACC_HST	Duration of a table walk for final guest translation, but limited to the duration only in level-2 host memory accesses and TLB/PWC reload.
Special	000000B0AA	PM_RDXWALK_TW0_FIN_GST_LVL3_ACC_HST	Duration of a table walk for final guest translation, but limited to the duration only in level-3 host memory accesses and TLB/PWC reload.
Special	000000B8AA	PM_RDXWALK_TW1_FIN_GST_LVL3_ACC_HST	Duration of a table walk for final guest translation, but limited to the duration only in level-3 host memory accesses and TLB/PWC reload.
Special	000000B0AC	PM_RDXWALK_TW0_FIN_GST_LVL4_ACC_HST	Duration of a table walk for final guest translation, but limited to the duration only in level-4 host memory accesses and TLB/PWC reload.
Special	000000B8AC	PM_RDXWALK_TW1_FIN_GST_LVL4_ACC_HST	Duration of a table walk for final guest translation, but limited to the duration only in level-4 host memory accesses and TLB/PWC reload.

5.12 Transactional Memory Events

Table 5-15 lists the transactional memory events.

Table 5-15. Transactional Memory Events (Sheet 1 of 2)

PMC	Event Code	Event Name	Event Description
PMC4	000004E014	PM_TM_TX_PASS_RUN_INST	Run instructions spent in successful transactions.
Any PMC	0000002094	PM_TM_OUTER_TBEGIN	Completion time outer **tbegin**.
Any PMC	0000002894	PM_TM_OUTER_TEND	Completion time outer **tend**.
Any PMC	0000002098	PM_TM_NESTED_TEND	Completion time nested **tend**.
Any PMC	0000002898	PM_TM_TABORT_TRECLAIM	Completion time **tabortnoncd**, **tabortcd**, **treclaim**.
Any PMC	000000209C	PM_TM_FAV_TBEGIN	Dispatch time favored **tbegin**.
Any PMC	000000289C	PM_TM_NON_FAV_TBEGIN	Dispatch time non-favored **tbegin**.
Any PMC	00000020A0	PM_TM_NESTED_TBEGIN	Completion time nested **tbegin**.
Any PMC	00000028A0	PM_TM_TSUSPEND	TM suspend instruction completed.
Any PMC	00000020A4	PM_TM_TRESUME	TM resume instruction completed.
Any PMC	00000028A4	PM_MRK_TEND_FAIL	Nested or not nested **tend** failed for a marked **tend** instruction.
Any PMC	00000020A8	PM_TM_FAIL_FOOTPRINT_OVERFLOW	TM aborted because the tracking limit for transactional storage accesses was exceeded. Asynchronous.
Any PMC	00000028A8	PM_TM_FAIL_CONF_NON_TM	TM aborted because a conflict occurred with a non-transactional access by another processor.
Any PMC	00000020AC	PM_TM_FAIL_CONF_TM	TM aborted because a conflict occurred with another transaction.
Any PMC	00000028AC	PM_TM_FAIL_SELF	TM aborted because a self-induced conflict occurred in a suspended state due to one of the following: a store to a storage location that was previously accessed transactionally; a **dcbf**, **dcbi**, or **icbi** specifying a block that was previously accessed transactionally; a **dcbst** specifying a block that was previously written transactionally; or a **tlbie** that specifies a translation that was a previously used transaction.
PMC2	000002E052	PM_TM_PASSED	Number of TM transactions that passed.
PMC3	0000030056	PM_TM_ABORTS	Number of TM transactions aborted.
PMC1	0000010060	PM_TM_TRANS_RUN_CYC	Number of run cycles in a transactional state.
PMC2	000002E012	PM_TM_TX_PASS_RUN_CYC	Number of cycles spent in successful transactions.
PMC4	000004E05E	PM_TM_OUTER_TBEGIN_DISP	Number of outer **tbegin** instructions dispatched. The dispatch unit determines whether the **tbegin** instruction is outer or nested. This is a speculative count that includes flushed instructions.
Any PMC	000000E0A4	PM_TMA_REQ_L2	Address only requests to the L2 cache. Multiple requests to the same line are only reported once. Indicates that the load footprint is not expanding.
Any PMC	000000E0AC	PM_TM_FAIL_TLBIE	Transaction failed because there was a TLBIE hit in the bloom filter.
Any PMC	000000E8AC	PM_TM_FAIL_TX_CONFLICT	Transactional conflict from the LSU, which is reported in the TEXASR.

Table 5-15. Transactional Memory Events (Sheet 2 of 2)

PMC	Event Code	Event Name	Event Description
Any PMC	000000E0B0	PM_TM_FAIL_NON_TX_CONFLICT	LSU reports the highest priority it reported to the ISU.
Any PMC	000000E8B0	PM_TEND_PEND_CYC	Number of cycles **tend** was waiting on a response.
Any PMC	000000E0B4	PM_LS0_TM_DISALLOW	Valid only for the master slice. One count per PPC, even if the instruction spans multiple slices.
Any PMC	000000E8B4	PM_LS1_TM_DISALLOW	Valid only for master slice. One count per PPC, even if the instruction spans multiple slices.
Any PMC	000000E0B8	PM_LS2_TM_DISALLOW	Valid only for master slice. One count per PPC, even if the instruction spans multiple slices.
Any PMC	000000E8B8	PM_LS3_TM_DISALLOW	Valid only for master slice. One count per PPC, even if the instruction spans multiple slices.

5.13 PMC Events

Table 5-16 lists the PMC events.

Table 5-16. PMC Events (Sheet 1 of 2)

PMC	Event Code	Event Name	Event Description
PMC1	0000010000	PM_SUSPENDED	Counter off.
PMC1	0000010008	PM_RUN_SPURR	Run SPURR.
PMC1	000001000A	PM_PMC3_REWIND	PMC3 rewind event. A rewind happens when a speculative event (such as latency or a CPI stack) is selected on PMC3 and the stall reason or reload source does not match the one programmed in PMC3. When this occurs, the count in PMC3 does not change.
PMC1	0000010010	PM_PMC4_OVERFLOW	Overflow from counter 4.
PMC1	0000010020	PM_PMC4_REWIND	PMC4 rewind event.
PMC1	0000010022	PM_PMC2_SAVED	PMC2 rewind value saved.
PMC1	0000010024	PM_PMC5_OVERFLOW	Overflow from counter 5.
PMC1	00000101E6	PM_THRESH_EXC_4096	Threshold counter exceed a count of 4096.
PMC1	00000101E8	PM_THRESH_EXC_256	Threshold counter exceed a count of 256.
PMC1	00000101EC	PM_THRESH_MET	Threshold exceeded.
PMC2	0000020000	PM_SUSPENDED	Counter off.
PMC2	0000020010	PM_PMC1_OVERFLOW	Overflow from counter 1.
PMC2	0000024154	PM_THRESH_ACC	This event increments every time the threshold event counter ticks. Thresholding must be enabled (via MMCRA) and the thresholding start event must occur for this counter to increment. It will stop incrementing when the thresholding stop event occurs or when thresholding is disabled, until the next time a configured thresholding start event occurs.
PMC2	00000201E6	PM_THRESH_EXC_32	Threshold counter exceeded a value of 32.
PMC2	00000201E8	PM_THRESH_EXC_512	Threshold counter exceeded a value of 512.
PMC3	0000030000	PM_SUSPENDED	Counter off.
PMC3	0000030010	PM_PMC2_OVERFLOW	Overflow from counter 2.

Table 5-16. PMC Events (Sheet 2 of 2)

PMC	Event Code	Event Name	Event Description
PMC3	0000030020	PM_PMC2_REWIND	PMC2 rewind event (did not match condition).
PMC3	0000030022	PM_PMC4_SAVED	PMC4 rewind value saved (matched condition).
PMC3	0000030024	PM_PMC6_OVERFLOW	Overflow from counter 6.
PMC3	00000301E8	PM_THRESH_EXC_64	Threshold counter exceeded a value of 64.
PMC3	00000301EA	PM_THRESH_EXC_1024	Threshold counter exceeded a value of 1024.
PMC4	0000040000	PM_SUSPENDED	Counter off.
PMC4	0000040010	PM_PMC3_OVERFLOW	Overflow from counter 3.
PMC4	000004D010	PM_PMC1_SAVED	PMC1 rewind value saved.
PMC4	000004D012	PM_PMC3_SAVED	PMC3 rewind value saved.
PMC4	000004D02C	PM_PMC1_REWIND	PMC1 rewind event.
PMC4	0000040062	PM_DUMMY1_REMOVE_ME	Space holder for L2_PC_PM_MK_LDST_SCOPE_PRED_STATUS.
PMC4	0000040064	PM_DUMMY2_REMOVE_ME	Space holder for LS_PC_RELOAD_RA.
PMC4	000004016E	PM_THRESH_NOT_MET	Threshold counter did not meet threshold.
PMC4	00000401EA	PM_THRESH_EXC_128	Threshold counter exceeded a value of 128.
PMC4	00000401EC	PM_THRESH_EXC_2048	Threshold counter exceeded a value of 2048.
PMC4	00000400F4	PM_RUN_PURR	Run PURR.
PMC1	000001001E	PM_CYC	Processor cycles.
PMC1	00000100F0	PM_CYC	Cycles.
PMC1	00000100FA	PM_ANY_THRD_RUN_CYC	Cycles in which at least one thread has the run latch set.
PMC2	000002000A	PM_HV_CYC	Cycles in which MSR[HV] is high. **Note:** This event does not take MSR[PR] into consideration.
PMC2	000002000C	PM_THRD_ALL_RUN_CYC	Cycles in which all the threads have the run latch set.
PMC2	000002001E	PM_CYC	Cycles.
PMC2	000002006C	PM_RUN_CYC_SMT4_MODE	Cycles in which this thread's run latch is set and the core is in SMT4 mode.
PMC2	00000200F4	PM_RUN_CYC	Run cycles.
PMC2	00000200F8	PM_EXT_INT	External interrupt.
PMC3	000003000C	PM_FREQ_DOWN	Power management: below threshold B.
PMC3	000003001E	PM_CYC	Cycles.
PMC3	000003006E	PM_NEST_REF_CLK	Multiply by 4 to obtain the number of SMP interconnect cycles.
PMC3	00000300F8	PM_TB_BIT_TRANS	Timebase even.
PMC4	000004000C	PM_FREQ_UP	Power management: above threshold A.
PMC4	0000040014	PM_PROBE_NOP_DISP	ProbeNops dispatched.
PMC4	000004001E	PM_CYC	Cycles.

5.14 Metrics

Table 5-17 through *Table 5-22* on page 101 list the metrics by category and *Table 5-23* on page 103 lists the detailed POWER9 metric events and formulas.

Table 5-17 list the general metrics.

Table 5-17. POWER9 Metrics (General)

Category	Metric
General	CPI
General	IPC
General	RUN_CPI
General	Run_Cycles(%)
General	Run_Latch_Cyc(%)
General	Speculation
General	L1_LD_Miss_Ratio(%)
General	L1_ST_Miss_Ratio(%)
General	L1_ST_Miss_Rate(%)
General	L1_LD_Miss_Rate(%)
General	L2_LD_Miss_Rate(%)
General	L3_LD_Miss_Rate(%)
General	L1_Inst_Miss_Rate(%)
General	L2_Inst_Miss_Rate(%)
General	L3_Inst_Miss_Rate(%)
General	DTLB_Miss_Rate(%)
General	ITLB_Miss_Rate(%)
General	ICACHE_PREF(%)
General	DERAT_Miss_Rate(%)
General	L2_PTEG_Miss_Rate(%)
General	L3_PTEG_Miss_Rate(%)
General	Flush_Rate(%)
General	Disp_Flush_Rate(%)
General	Br_Mpred_Flush_Rate(%)
General	Avg_LMQ_Life_Time
General	Avg_LRQ_Life_Time_Even
General	Avg_LRQ_Life_Time_Odd
General	Avg_SRQ_Life_Time_Even
General	Avg_SRQ_Life_Time_Odd

Table 5-18 lists the metrics for the CPI breakdown category.

Table 5-18. POWER9 Metrics (CPI Breakdown) (Sheet 1 of 3)

Category	Metric
CPI_Breakdown	RUN_CPI
CPI_Breakdown	LRQ_OTHER_STALL_CPI
CPI_Breakdown	THRD_STALL_CPI
CPI_Breakdown	LARX_STALL_CPI
CPI_Breakdown	LSU_FIN_STALL_CPI
CPI_Breakdown	DMISS_L2L3_STALL_CPI
CPI_Breakdown	DFLONG_STALL_CPI
CPI_Breakdown	DP_STALL_CPI
CPI_Breakdown	TEND_STALL_CPI
CPI_Breakdown	SLB_STALL_CPI
CPI_Breakdown	STALL_CPI
CPI_Breakdown	FLUSH_ANY_THREAD_STALL_CPI
CPI_Breakdown	SYS_CALL_STALL_CPI
CPI_Breakdown	NESTED_TBEGIN_STALL_CPI
CPI_Breakdown	LSU_STALL_CPI
CPI_Breakdown	DCACHE_MISS_STALL_CPI
CPI_Breakdown	STORE_FINISH_STALL_CPI
CPI_Breakdown	PASTE_STALL_CPI
CPI_Breakdown	DMISS_L21_L31_STALL_CPI
CPI_Breakdown	LHS_STALL_CPI
CPI_Breakdown	DMISS_REMOTE_STALL_CPI
CPI_Breakdown	RFID_STALL_CPI
CPI_Breakdown	DFU_STALL_CPI
CPI_Breakdown	LRQ_FULL_STALL_CPI
CPI_Breakdown	FXU_STALL_CPI
CPI_Breakdown	EXEC_UNIT_STALL_CPI
CPI_Breakdown	EXEC_UNIT_OTHER_STALL_CPI
CPI_Breakdown	STCX_STALL_CPI
CPI_Breakdown	VFXLONG_STALL_CPI
CPI_Breakdown	LSU_FLUSH_NEXT_STALL_CPI
CPI_Breakdown	TLBIE_STALL_CPI
CPI_Breakdown	NTC_FLUSH_STALL_CPI
CPI_Breakdown	EMQ_FULL_STALL_CPI
CPI_Breakdown	PM_STALL_CPI
CPI_Breakdown	STORE_FIN_ARB_STALL_CPI

Table 5-18. POWER9 Metrics (CPI Breakdown) (Sheet 2 of 3)

Category	Metric
CPI_Breakdown	SRQ_FULL_STALL_CPI
CPI_Breakdown	STORE_DATA_STALL_CPI
CPI_Breakdown	SPEC_FINISH_STALL_CPI
CPI_Breakdown	DMISS_LMEM_STALL_CPI
CPI_Breakdown	EXCEPTION_STALL_CPI
CPI_Breakdown	NESTED_TEND_STALL_CPI
CPI_Breakdown	VDPLONG_STALL_CPI
CPI_Breakdown	VFXU_STALL_CPI
CPI_Breakdown	LSU_MFSPR_STALL_CPI
CPI_Breakdown	DARQ_STALL_CPI
CPI_Breakdown	DPLONG_STALL_CPI
CPI_Breakdown	STORE_PIPE_ARB_STALL_CPI
CPI_Breakdown	ERAT_MISS_STALL_CPI
CPI_Breakdown	LMQ_FULL_STALL_CPI
CPI_Breakdown	DMISS_L2L3_CONFLICT_STALL_CPI
CPI_Breakdown	DMISS_L3MISS_STALL_CPI
CPI_Breakdown	ST_FWD_STALL_CPI
CPI_Breakdown	CRYPTO_STALL_CPI
CPI_Breakdown	LOAD_FINISH_STALL_CPI
CPI_Breakdown	FXLONG_STALL_CPI
CPI_Breakdown	BRU_STALL_CPI
CPI_Breakdown	EIEIO_STALL_CPI
CPI_Breakdown	MTFPSCR_STALL_CPI
CPI_Breakdown	LSAQ_ARB_STALL_CPI
CPI_Breakdown	NTC_DISP_FIN_STALL_CPI
CPI_Breakdown	VDP_STALL_CPI
CPI_Breakdown	ICT_NOSLOT_DISP_HELD_TBEGIN_CPI
CPI_Breakdown	NTC_ISSUE_HELD_DARQ_FULL_CPI
CPI_Breakdown	ICT_NOSLOT_CYC_CPI
CPI_Breakdown	ICT_NOSLOT_IC_MISS_CPI
CPI_Breakdown	ICT_NOSLOT_DISP_HELD_ISSQ_CPI
CPI_Breakdown	NTC_ISSUE_HELD_ARB_CPI
CPI_Breakdown	NTC_FIN_CPI
CPI_Breakdown	RUN_CYC_CPI
CPI_Breakdown	ICT_NOSLOT_DISP_HELD_HB_FULL_CPI
CPI_Breakdown	ICT_NOSLOT_BR_MPRED_ICMISS_CPI
CPI_Breakdown	NTC_ISSUE_HELD_OTHER_CPI

Table 5-18. POWER9 Metrics (CPI Breakdown) (Sheet 3 of 3)

Category	Metric
CPI_Breakdown	ICT_NOSLOT_IC_L3_CPI
CPI_Breakdown	ICT_NOSLOT_DISP_HELD_SYNC_CPI
CPI_Breakdown	ICT_NOSLOT_BR_MPRED_CPI
CPI_Breakdown	ICT_NOSLOT_IC_L3MISS_CPI
CPI_Breakdown	ICT_NOSLOT_DISP_HELD_CPI
CPI_Breakdown	ICT_NOSLOT_IC_L2_CPI
CPI_Breakdown	ICT_NOSLOT_DISP_HELD_OTHER_CPI
CPI_Breakdown	ISSUE_HOLD_CPI
CPI_Breakdown	SCALAR_STALL_CPI
CPI_Breakdown	VECTOR_STALL_CPI
CPI_Breakdown	LSAQ_STALL_CPI
CPI_Breakdown	EMQ_STALL_CPI
CPI_Breakdown	LRQ_STALL_CPI
CPI_Breakdown	SRQ_STALL_CPI
CPI_Breakdown	ICT_NOSLOT_CYC_OTHER_CPI
CPI_Breakdown	FXU_OTHER_STALL_CPI
CPI_Breakdown	DP_OTHER_STALL_CPI
CPI_Breakdown	DFU_OTHER_STALL_CPI
CPI_Breakdown	VFXU_OTHER_STALL_CPI
CPI_Breakdown	VDP_OTHER_STALL_CPI
CPI_Breakdown	DMISS_L2L3_NOCONFLICT_STALL_CPI
CPI_Breakdown	DMISS_DMEM_STALL_CPI
CPI_Breakdown	LSU_OTHER_STALL_CPI
CPI_Breakdown	OTHER_STALL_CPI
CPI_Breakdown	OTHER_CPI

Table 5-19 lists the cache-related metrics.

Table 5-19. POWER9 Metrics (Cache) (Sheet 1 of 3)

Category	Metric
Data L1 Cache Reloads (% per ref)	dL1_Miss_Reloads(%)
Data L1 Cache Reloads (% per ref)	dL1_Reload_FROM_L2(%)
Data L1 Cache Reloads (% per ref)	dL1_Reload_FROM_L21_MOD(%)
Data L1 Cache Reloads (% per ref)	dL1_Reload_FROM_L21_SHR(%)
Data L1 Cache Reloads (% per ref)	dL1_Reload_FROM_L31_MOD(%)
Data L1 Cache Reloads (% per ref)	dL1_Reload_FROM_L31_SHR(%)
Data L1 Cache Reloads (% per ref)	dL1_Reload_FROM_L3(%)

Table 5-19. POWER9 Metrics (Cache) (Sheet 2 of 3)

Category	Metric
Data L1 Cache Reloads (% per ref)	dL1_Reload_FROM_RL2L3_SHR(%)
Data L1 Cache Reloads (% per ref)	dL1_Reload_FROM_RL2L3_MOD(%)
Data L1 Cache Reloads (% per ref)	dL1_Reload_FROM_DL2L3_MOD(%)
Data L1 Cache Reloads (% per ref)	dL1_Reload_FROM_DL2L3_SHR(%)
Data L1 Cache Reloads (% per ref)	dL1_Reload_FROM_LL4(%)
Data L1 Cache Reloads (% per ref)	dL1_Reload_FROM_LMEM(%)
Data L1 Cache Reloads (% per ref)	dL1_Reload_FROM_RL4(%)
Data L1 Cache Reloads (% per ref)	dL1_Reload_FROM_RMEM(%)
Data L1 Cache Reloads (% per ref)	dL1_Reload_FROM_DL4(%)
Data L1 Cache Reloads (% per ref)	dL1_Reload_FROM_DMEM(%)
Estimated Data Cache Miss CPI	dcache_miss_cpi(%)
Estimated Data Cache Miss CPI	L2_cpi(%)
Estimated Data Cache Miss CPI	L21_MOD_cpi(%)
Estimated Data Cache Miss CPI	L21_SHR_cpi(%)
Estimated Data Cache Miss CPI	L31_MOD_cpi(%)
Estimated Data Cache Miss CPI	L31_SHR_cpi(%)
Estimated Data Cache Miss CPI	L3_cpi(%)
Estimated Data Cache Miss CPI	RL2L3_SHR_cpi(%)
Estimated Data Cache Miss CPI	RL2L3_MOD_cpi(%)
Estimated Data Cache Miss CPI	DL2L3_MOD_cpi(%)
Estimated Data Cache Miss CPI	DL2L3_SHR_cpi(%)
Estimated Data Cache Miss CPI	LMEM_cpi(%)
Estimated Data Cache Miss CPI	RL4_cpi(%)
Estimated Data Cache Miss CPI	RMEM_cpi(%)
Estimated Data Cache Miss CPI	DL4_cpi(%)
Estimated Data Cache Miss CPI	DMEM_cpi(%)
Latency	Average_iL1_Miss_Latency
Latency	Estimated_dL1Miss_Latency
Latency	Exposed_dL1Miss_Latency
Latency	L2_Latency
Latency	L21_MOD_Latency
Latency	L21_SHR_Latency
Latency	L31_MOD_Latency
Latency	L31_SHR_Latency
Latency	L3_Latency
Latency	RL2L3_SHR_Latency
Latency	RL2L3_MOD_Latency

Table 5-19. POWER9 Metrics (Cache) (Sheet 3 of 3)

Category	Metric
Latency	DL2L3_MOD_Latency
Latency	DL2L3_SHR_Latency
Latency	LL4_Latency
Latency	LMEM_Latency
Latency	RL4_Latency
Latency	RMEM_Latency
Latency	DL4_Latency
Latency	DMEM_Latency
Latency	Avg_LMQ_Life_Time
Latency	Avg_LRQ_Life_Time_Even
Latency	Avg_LRQ_Life_Time_Odd
Latency	Avg_SRQ_Life_Time_Even
Latency	Avg_SRQ_Life_Time_Odd
Data L1 Cache Reloads (% per instruction)	L1_LD_Miss_Rate(%)
Data L1 Cache Reloads (% per instruction)	dL1_Reload_FROM_L2_Rate(%)
Data L1 Cache Reloads (% per instruction)	dL1_Reload_FROM_L21_MOD_Rate(%)
Data L1 Cache Reloads (% per instruction)	dL1_Reload_FROM_L21_SHR_Rate(%)
Data L1 Cache Reloads (% per instruction)	dL1_Reload_FROM_L31_MOD_Rate(%)
Data L1 Cache Reloads (% per instruction)	dL1_Reload_FROM_L31_SHR_Rate(%)
Data L1 Cache Reloads (% per instruction)	dL1_Reload_FROM_L2_Miss_Rate(%)
Data L1 Cache Reloads (% per instruction)	dL1_Reload_FROM_L3_Rate(%)
Data L1 Cache Reloads (% per instruction)	dL1_Reload_FROM_L3_Miss_Rate(%)
Data L1 Cache Reloads (% per instruction)	dL1_Reload_FROM_RL2L3_SHR_Rate(%)
Data L1 Cache Reloads (% per instruction)	dL1_Reload_FROM_RL2L3_MOD_Rate(%)
Data L1 Cache Reloads (% per instruction)	dL1_Reload_FROM_DL2L3_MOD_Rate(%)
Data L1 Cache Reloads (% per instruction)	dL1_Reload_FROM_DL2L3_SHR_Rate(%)
Data L1 Cache Reloads (% per instruction)	dL1_Reload_FROM_LL4_Rate(%)
Data L1 Cache Reloads (% per instruction)	dL1_Reload_FROM_LMEM_Rate(%)
Data L1 Cache Reloads (% per instruction)	dL1_Reload_FROM_RL4_Rate(%)
Data L1 Cache Reloads (% per instruction)	dL1_Reload_FROM_RMEM_Rate(%)
Data L1 Cache Reloads (% per instruction)	dL1_Reload_FROM_DL4_Rate(%)
Data L1 Cache Reloads (% per instruction)	dL1_Reload_FROM_DMEM_Rate(%)

Table 5-20 lists the memory-related metrics.

Table 5-20. POWER9 Metrics (Memory)

Category	Metric
Memory	MEM_LOCALITY(%)
Memory	LD_LMEM_PER_LD_RMEM
Memory	LD_LMEM_PER_LD_DMEM
Memory	LD_LMEM_PER_LD_MEM
Memory	LD_RMEM_PER_LD_DMEM
Memory	L4_LOCALITY
Memory	LD_LL4_PER_LD_RL4
Memory	LD_LL4_PER_LD_DMEM
Memory	LD_LL4_PER_LD_MEM

Table 5-21 lists translation-related metrics.

Table 5-21. POWER9 Metrics (Translation) (Sheet 1 of 3)

Category	Metric
Translation	DERAT_Miss_Rate(%)
Translation	DERAT_4K_Miss_Rate(%)
Translation	DERAT_64K_Miss_Rate(%)
Translation	DERAT_16M_Miss_Rate(%)
Translation	DERAT_16G_Miss_Rate(%)
Translation	DSLB_Miss_Rate(%)
Translation	DERAT_Miss_Ratio
Translation	DERAT_4K_Miss_Ratio
Translation	DERAT_64K_Miss_Ratio
Translation	DERAT_16M_Miss_Ratio
Translation	DERAT_16G_Miss_Ratio
Translation	L1_Inst_Miss_Rate(%)
Translation	IERAT_Miss_Rate(%)
Translation	ISLB_Miss_Rate(%)
PTEG Reloads (% per ref)	DERAT_MISS_RELOAD(%)
PTEG Reloads (% per ref)	PTEG_FROM_L2(%)
PTEG Reloads (% per ref)	PTEG_FROM_L21_MOD(%)
PTEG Reloads (% per ref)	PTEG_FROM_L21_SHR(%)
PTEG Reloads (% per ref)	PTEG_FROM_L31_MOD(%)
PTEG Reloads (% per ref)	PTEG_FROM_L31_SHR(%)
PTEG Reloads (% per ref)	PTEG_FROM_L3(%)
PTEG Reloads (% per ref)	PTEG_FROM_RL2L3_SHR(%)

Table 5-21. POWER9 Metrics (Translation) (Sheet 2 of 3)

Category	Metric
PTEG Reloads (% per ref)	PTEG_FROM_RL2L3_MOD(%)
PTEG Reloads (% per ref)	PTEG_FROM_DL2L3_MOD(%)
PTEG Reloads (% per ref)	PTEG_FROM_DL2L3_SHR(%)
PTEG Reloads (% per ref)	PTEG_FROM_LL4(%)
PTEG Reloads (% per ref)	PTEG_FROM_LMEM(%)
PTEG Reloads (% per ref)	PTEG_FROM_RL4(%)
PTEG Reloads (% per ref)	PTEG_FROM_RMEM(%)
PTEG Reloads (% per ref)	PTEG_FROM_DL4(%)
PTEG Reloads (% per ref)	PTEG_FROM_DMEM(%)
PTEG Reloads (% per instruction)	DERAT_Miss_Rate(%)
PTEG Reloads (% per instruction)	PTEG_FROM_L2_Rate(%)
PTEG Reloads (% per instruction)	PTEG_FROM_L21_MOD_Rate(%)
PTEG Reloads (% per instruction)	PTEG_FROM_L21_SHR_Rate(%)
PTEG Reloads (% per instruction)	PTEG_FROM_L31_MOD_Rate(%)
PTEG Reloads (% per instruction)	PTEG_FROM_L31_SHR_Rate(%)
PTEG Reloads (% per instruction)	PTEG_FROM_L3_Rate(%)
PTEG Reloads (% per instruction)	PTEG_FROM_RL2L3_SHR_Rate(%)
PTEG Reloads (% per instruction)	PTEG_FROM_RL2L3_MOD_Rate(%)
PTEG Reloads (% per instruction)	PTEG_FROM_DL2L3_MOD_Rate(%)
PTEG Reloads (% per instruction)	PTEG_FROM_DL2L3_SHR_Rate(%)
PTEG Reloads (% per instruction)	PTEG_FROM_LL4_Rate(%)
PTEG Reloads (% per instruction)	PTEG_FROM_LMEM_Rate(%)
PTEG Reloads (% per instruction)	PTEG_FROM_RL4_Rate(%)
PTEG Reloads (% per instruction)	PTEG_FROM_RMEM_Rate(%)
PTEG Reloads (% per instruction)	PTEG_FROM_DL4_Rate(%)
PTEG Reloads (% per instruction)	PTEG_FROM_DMEM_Rate(%)
LSU Rejects	LSU_Reject_Ratio(%)
LSU Rejects	LHS_Reject_Ratio(%)
LSU Rejects	ERAT_Reject_Ratio(%)
LSU Rejects	LMQ_full_Reject_Ratio(%)
LSU Rejects	Set_Mpred_Reject_Ratio(%)
LSU Rejects	LSU_Reject_Rate(%)
LSU Rejects	LHS_Reject_Rate(%)
LSU Rejects	ERAT_Reject_Rate(%)
LSU Rejects	LMQ_full_Reject_Rate(%)
LSU Rejects	Set_Mpred_Reject_Rate(%)
Instruction Misses (% per inst)	L1_Inst_Miss_Rate(%)

Table 5-21. POWER9 Metrics (Translation) (Sheet 3 of 3)

Category	Metric
Instruction Misses (% per inst)	INST_FROM_L2_Rate(%)
Instruction Misses (% per inst)	INST_FROM_L21_MOD_Rate(%)
Instruction Misses (% per inst)	INST_FROM_L21_SHR_Rate(%)
Instruction Misses (% per inst)	INST_FROM_L31_MOD_Rate(%)
Instruction Misses (% per inst)	INST_FROM_L31_SHR_Rate(%)
Instruction Misses (% per inst)	INST_FROM_L3_Rate(%)
Instruction Misses (% per inst)	INST_FROM_RL2L3_SHR_Rate(%)
Instruction Misses (% per inst)	INST_FROM_RL2L3_MOD_Rate(%)
Instruction Misses (% per inst)	INST_FROM_DL2L3_MOD_Rate(%)
Instruction Misses (% per inst)	INST_FROM_DL2L3_SHR_Rate(%)
Instruction Misses (% per inst)	INST_FROM_LL4_Rate(%)
Instruction Misses (% per inst)	INST_FROM_LMEM_Rate(%)
Instruction Misses (% per inst)	INST_FROM_RL4_Rate(%)
Instruction Misses (% per inst)	INST_FROM_RMEM_Rate(%)
Instruction Misses (% per inst)	INST_FROM_DL4_Rate(%)
Instruction Misses (% per inst)	INST_FROM_DMEM_Rate(%)
Prefetch	L1_Prefetch_Rate(%)
Prefetch	L3_LD_Prefetch_Rate(%)
Prefetch	L3_ST_Prefetch_Rate(%)
Prefetch	L3_LDST_Prefetch_Rate(%)
Prefetch	Prefetch_stream_Alloc_per_Confirm
Prefetch	Strided_Prefetch_stream_Alloc_per_Confirm

Table 5-22 lists statistic-related metrics for a variety categories.

Table 5-22. POWER9 Metrics (Statistics) (Sheet 1 of 3)

Category	Metric
Instruction Statistics (% per ref)	ICACHE_PREF(%)
Instruction Statistics (% per ref)	ICache_MISS_RELOAD
Instruction Statistics (% per ref)	INST_FROM_L2(%)
Instruction Statistics (% per ref)	INST_FROM_L21_MOD(%)
Instruction Statistics (% per ref)	INST_FROM_L21_SHR(%)
Instruction Statistics (% per ref)	INST_FROM_L31_MOD(%)
Instruction Statistics (% per ref)	INST_FROM_L31_SHR(%)
Instruction Statistics (% per ref)	INST_FROM_L3(%)
Instruction Statistics (% per ref)	INST_FROM_RL2L3_SHR(%)
Instruction Statistics (% per ref)	INST_FROM_RL2L3_MOD(%)

Table 5-22. POWER9 Metrics (Statistics) (Sheet 2 of 3)

Category	Metric
Instruction Statistics (% per ref)	INST_FROM_DL2L3_MOD(%)
Instruction Statistics (% per ref)	INST_FROM_DL2L3_SHR(%)
Instruction Statistics (% per ref)	INST_FROM_LL4(%)
Instruction Statistics (% per ref)	INST_FROM_LMEM(%)
Instruction Statistics (% per ref)	INST_FROM_RL4(%)
Instruction Statistics (% per ref)	INST_FROM_RMEM(%)
Instruction Statistics (% per ref)	INST_FROM_DL4(%)
Instruction Statistics (% per ref)	INST_FROM_DMEM(%)
L2 Cache Statistics	L2_LD_Disp(%)
L2 Cache Statistics	L2_ST_Disp(%)
L2 Cache Statistics	L2_INST_Disp(%)
L2 Cache Statistics	L2_LD_Miss_Ratio(%)
L2 Cache Statistics	L2_ST_Miss_Ratio(%)
L2 Cache Statistics	L2_ld_hit_frequency
L2 Cache Statistics	L2_ld_miss_frequency
L2 Cache Statistics	L2_INST_Miss_Ratio(%)
L2 Cache Statistics	L2_RC_LD_Disp_Fail(%)
L2 Cache Statistics	L2_RC_LD_Disp_Addr_Fail(%)
L2 Cache Statistics	L2_RC_ST_Disp_Fail(%)
L2 Cache Statistics	L2_RC_ST_Disp_Addr_Fail(%)
L2 Cache Statistics	RC_LD_Busy(%)
L2 Cache Statistics	RC_ST_Busy(%)
L2 Cache Statistics	L2_Node_Pumps(%)
L2 Cache Statistics	L2_Sys_Pumps(%)
L2 Cache Statistics	L2_Global_Pred_Correct(%)
L2 Cache Statistics	L2_Local_Pred_Correct(%)
L2 Cache Statistics	L2_IC_Inv_Rate(%)
L2 Cache Statistics	L2_DC_Inv_Rate(%)
L2 Cache Statistics	L2_Dem_LD_Disp(%)
L2 Cache Statistics	L2_Mod_CO(%)
L2 Cache Statistics	L2_Shr_CO(%)
L2 Cache Statistics	L2_LD_Rd_Util
L2 Cache Statistics	L2_ST_Rd_Util
L2 Cache Statistics	L2_CO_M_Rd_Util
L2 Cache Statistics	L2_Rd_Util(%)
L2 Cache Statistics	L2_LDMISS_Wr_Util
L2 Cache Statistics	L2_ST_Wr_Util

Table 5-22. POWER9 Metrics (Statistics) (Sheet 3 of 3)

Category	Metric
L2 Cache Statistics	L2_Wr_Util(%)
L3 Cache Statistics	L3_ld_hit_frequency
L3 Cache Statistics	L3_ld_miss_frequency
L3 Cache Statistics	L3_RD_Lifetime
L3 Cache Statistics	L3_PF_Lifetime
L3 Cache Statistics	L3_SN_Lifetime
L3 Cache Statistics	L3_CO_Lifetime
L3 Cache Statistics	L3_WI_Lifetime
L3 Cache Statistics	L3_WI_Usage

Table 5-23 defines the POWER9 metric events.

Table 5-23. POWER9 Metric Events and Formulas (Sheet 1 of 23)

Event Name	Event Description	Formula	Unit
Speculation	Instruction dispatch-to-completion ratio.	PM_INST_DISP / PM_INST_CMPL	
Average Completed Instruction Set Size	Average more than one instruction completed.	PM_INST_CMPL / PM_1PLUS_PPC_CMPL	
L1_LD_Miss_Ratio(%)	Percentage of L1 load misses per L1 load reference.	PM_LD_MISS_L1 / PM_LD_REF_L1 * 100	%
L1_LD_Miss_Rate(%)	Percentage of L1 demand load misses per run instruction.	PM_LD_MISS_L1 * 100 / PM_RUN_INST_CMPL	%
L1_ST_Miss_Rate(%)	Percentage of L1 store misses per run instruction.	PM_ST_MISS_L1 * 100 / PM_RUN_INST_CMPL	%
L2_LD_Miss_Rate(%)	L2 demand load miss rate (per run instruction).	PM_DATA_FROM_L2MISS * 100 / PM_RUN_INST_CMPL	%
L3_LD_Miss_Rate(%)	L3 demand load miss rate (per run instruction).	PM_DATA_FROM_L3MISS * 100 / PM_RUN_INST_CMPL	%
DERAT_Miss_Rate(%)	DERAT miss rate (per run instruction).	PM_LSU_DERAT_MISS * 100 / PM_RUN_INST_CMPL	%
L2_PTEG_Miss_Rate(%)	L2 PTEG miss rate (per run instruction).	PM_DPTEG_FROM_L2MISS * 100 / PM_RUN_INST_CMPL	%
L3_PTEG_Miss_Rate(%)	L3 PTEG miss rate (per run instruction).	PM_DPTEG_FROM_L3MISS * 100 / PM_RUN_INST_CMPL	%
L1_Inst_Miss_Rate(%)	Instruction cache miss rate (per run instruction).	PM_L1_ICACHE_MISS * 100 / PM_RUN_INST_CMPL	%
ICACHE_PREF(%)	Percentage of I-cache reloads due to prefetch.	PM_IC_PREF_WRITE * 100 / PM_L1_ICACHE_MISS	%
L2_Inst_Miss_Rate(%)	L2 instruction miss rate (per instruction).	PM_INST_FROM_L2MISS * 100 / PM_RUN_INST_CMPL	%
L3_Inst_Miss_Rate(%)	L3 instruction miss rate (per instruction).	PM_INST_FROM_L3MISS * 100 / PM_RUN_INST_CMPL	%
CPI	Cycles per instruction.	PM_CYC / PM_INST_CMPL	
IPC	Instructions per cycles.	PM_INST_CMPL / PM_CYC	

Table 5-23. POWER9 Metric Events and Formulas (Sheet 2 of 23)

Event Name	Event Description	Formula	Unit
Cycles/Completed_Instructions_Set	Cycles per instruction group	PM_CYC / PM_1PLUS_PPC_CMPL	
Run_Cycles(%)	Run cycles per cycle.	PM_RUN_CYC / PM_CYC*100	
elapased_cycles	Time-base elapsed cycles.	proc_freq * total_time	
Run_Latch_Cyc(%)	Percentage of all cycles in which the run latch was set (assumes a fixed frequency).	(PM_RUN_CYC / elpased_cycles) * 100	%
Cycles_Atleast_One_Inst_Dispatched(%)	Percentage of cycles in which at least one instruction is dispatched.	PM_1PLUS_PPC_DISP / PM_CYC * 100	%
RUN_CPI	Run cycles per run instruction.	PM_RUN_CYC / PM_RUN_INST_CMPL	
LRQ_OTHER_STALL_CPI	Finish stall due to LRQ miscellaneous reasons, lost arbitration to LMQ slot, bank collisions, set prediction cleanup, set prediction multi-hit, and others.	PM_CMPLU_STALL_LRQ_OTHER/PM_RUN_INST_CMPL	
THRD_STALL_CPI	Completion stalled because the thread was blocked.	PM_CMPLU_STALL_THRD/PM_RUN_INST_CMPL	
LARX_STALL_CPI	Finish stall because the NTF instruction was a **larx** waiting to be satisfied.	PM_CMPLU_STALL_LARX/PM_RUN_INST_CMPL	
LSU_FIN_STALL_CPI	Finish stall because the NTF instruction was an LSU operation (other than a load or a store) with all of its dependencies met and only going through the LSU pipe to finish.	PM_CMPLU_STALL_LSU_FIN/PM_RUN_INST_CMPL	
DMISS_L2L3_STALL_CPI	Completion stall by a D-cache miss that resolved in the L2 or L3 cache.	PM_CMPLU_STALL_DMISS_L2L3/PM_RUN_INST_CMPL	
DFLONG_STALL_CPI	Finish stall because the NTF instruction was a multi-cycle instruction issued to the decimal floating-point execution pipe and is waiting to finish.	PM_CMPLU_STALL_DFLONG/PM_RUN_INST_CMPL	
DP_STALL_CPI	Finish stall because the NTF instruction was a scalar instruction issued to the double-precision execution pipe and is waiting to finish. Includes binary floating-point instructions in 32- and 64-bit binary floating-point format.	PM_CMPLU_STALL_DP/PM_RUN_INST_CMPL	
ANY_SYNC_STALL_CPI	ANY_SYNC_STALL_CPI.	PM_CMPLU_STALL_ANY_SYNC / PM_RUN_INST_CMPL	
SYNC_PMU_INT_STALL_CPI	ANY_SYNC_STALL_CPI.	PM_CMPLU_STALL_SYNC_PMU_INT / PM_RUN_INST_CMPL	
NTC_ALL_FIN_CPI	ANY_SYNC_STALL_CPI.	PM_NTC_ALL_FIN / PM_RUN_INST_CMPL	
TEND_STALL_CPI	Finish stall because the NTF instruction was a **tend** instruction waiting for a response from the L2 cache.	PM_CMPLU_STALL_TEND/PM_RUN_INST_CMPL	

Table 5-23. POWER9 Metric Events and Formulas (Sheet 3 of 23)

Event Name	Event Description	Formula	Unit
SLB_STALL_CPI	Finish stall because the NTF instruction was waiting for an L2 response for an SLB.	PM_CMPLU_STALL_SLB/PM_RUN_INST_CMPL	
STALL_CPI	Nothing completed and ICT is not empty.	PM_CMPLU_STALL/PM_RUN_INST_CMPL	
FLUSH_ANY_THREAD_STALL_CPI	Cycles in which the NTC instruction is not allowed to complete because any of the four threads in the same core suffered a flush, which blocks completion.	PM_CMPLU_STALL_FLUSH_ANY_THREAD/PM_RUN_INST_CMPL	
SYS_CALL_STALL_CPI	Cycles in which the NTC instruction is not allowed to complete because it was interrupted by a system call exception, which has to be serviced before the instruction can complete.	PM_CMPLU_STALL_SYS_CALL/PM_RUN_INST_CMPL	
NESTED_TBEGIN_STALL_CPI	Completion stall because the ISU is updating the TEXASR to keep track of the nested **tbegin**. This is a short delay and it includes ROT.	PM_CMPLU_STALL_NESTED_TBEGIN/PM_RUN_INST_CMPL	
LSU_STALL_CPI	Completion stall by an LSU instruction.	PM_CMPLU_STALL_LSU/PM_RUN_INST_CMPL	
DCACHE_MISS_STALL_CPI	Finish stall because the NTF instruction was a load that missed the L1 cache and was waiting for the data to return from the nest.	PM_CMPLU_STALL_DCACHE_MISS/PM_RUN_INST_CMPL	
STORE_FINISH_STALL_CPI	Finish stall because the NTF instruction was a store with all of its dependencies met, and is only waiting to go through the LSU pipe to finish.	PM_CMPLU_STALL_STORE_FINISH/PM_RUN_INST_CMPL	
PASTE_STALL_CPI	Finish stall because the NTF instruction was a paste waiting for a response from the L2 cache.	PM_CMPLU_STALL_PASTE/PM_RUN_INST_CMPL	
DMISS_L21_L31_STALL_CPI	Completion stall by a D-cache miss that resolved on chip (excluding the local L2 or L3 cache).	PM_CMPLU_STALL_DMISS_L21_L31/PM_RUN_INST_CMPL	
LHS_STALL_CPI	Finish stall because the NTF instruction was a load that hit on an older store and it was waiting for store data.	PM_CMPLU_STALL_LHS/PM_RUN_INST_CMPL	
DMISS_REMOTE_STALL_CPI	Completion stall by a D-cache miss that resolved from remote chip (cache or memory).	PM_CMPLU_STALL_DMISS_REMOTE/PM_RUN_INST_CMPL	
RFID_STALL_CPI	Cycles in which the NTC instruction is not allowed to complete because it was interrupted by an RFID exception that has to be serviced before the instruction can complete.	PM_CMPLU_STALL_RFID/PM_RUN_INST_CMPL	
DFU_STALL_CPI	Finish stall because the NTF instruction was issued to the decimal floating-point execution pipe and is waiting to finish.	PM_CMPLU_STALL_DFU/PM_RUN_INST_CMPL	

Table 5-23. POWER9 Metric Events and Formulas (Sheet 4 of 23)

Event Name	Event Description	Formula	Unit
LRQ_FULL_STALL_CPI	Finish stall because the NTF instruction was a load that was held in the LSAQ because the LRQ was full.	PM_CMPLU_STALL_LRQ_FULL/PM_RUN_INST_CMPL	
FXU_STALL_CPI	Finish stall due to a scalar fixed-point or CR instruction in the execution pipeline. These instructions are routed to the ALU, ALU2, and DIV pipes.	PM_CMPLU_STALL_FXU/PM_RUN_INST_CMPL	
EXEC_UNIT_STALL_CPI	Completion stall due to execution units (FXU/VSU/CRU).	PM_CMPLU_STALL_EXEC_UNIT/PM_RUN_INST_CMPL	
EXEC_UNIT_OTHER_STALL_CPI	Completion stall due to execution units for other reasons.	EXEC_UNIT_STALL_CPI-SCALAR_STALL_CPI-VECTOR_STALL_CPI	
STCX_STALL_CPI	Finish stall because the NTF instruction was a **stcx** waiting for a response from the L2 cache.	PM_CMPLU_STALL_STCX/PM_RUN_INST_CMPL	
VFXLONG_STALL_CPI	Completion stall due to a long latency vector fixed point instruction (division, square root)	PM_CMPLU_STALL_VFXLONG/PM_RUN_INST_CMPL	
LSU_FLUSH_NEXT_STALL_CPI	Completion stall of one cycle because the LSU requested to flush the next iop in the sequence. It takes one cycle for the ISU to process this request before the LSU instruction is allowed to complete.	PM_CMPLU_STALL_LSU_FLUSH_NEXT/PM_RUN_INST_CMPL	
TLBIE_STALL_CPI	Finish stall because the NTF instruction was a **tlbie** waiting for a response from the L2 cache.	PM_CMPLU_STALL_TLBIE/PM_RUN_INST_CMPL	
NTC_FLUSH_STALL_CPI	Completion stall due to a NTC flush.	PM_CMPLU_STALL_NTC_FLUSH/PM_RUN_INST_CMPL	
EMQ_FULL_STALL_CPI	Finish stall because the NTF instruction suffered an ERAT miss and the EMQ was full.	PM_CMPLU_STALL_EMQ_FULL/PM_RUN_INST_CMPL	
PM_STALL_CPI	Finish stall because the NTF instruction was issued to the Permute execution pipe and is waiting to finish.	PM_CMPLU_STALL_PM/PM_RUN_INST_CMPL	
STORE_FIN_ARB_STALL_CPI	Finish stall because the NTF instruction was a store waiting for a slot in the store finish pipe. This means the instruction is ready to finish, but there are instructions ahead of it using the finish pipe.	PM_CMPLU_STALL_STORE_FIN_ARB/PM_RUN_INST_CMPL	
SRQ_FULL_STALL_CPI	Finish stall because the NTF instruction was a store that was held in the LSAQ because the SRQ is full.	PM_CMPLU_STALL_SRQ_FULL/PM_RUN_INST_CMPL	
STORE_DATA_STALL_CPI	Finish stall because the NTF instruction is a store waiting on data.	PM_CMPLU_STALL_STORE_DATA/PM_RUN_INST_CMPL	

Table 5-23. POWER9 Metric Events and Formulas (Sheet 5 of 23)

Event Name	Event Description	Formula	Unit
SPEC_FINISH_STALL_CPI	Finish stall while waiting for the non-speculative finish of either a **stcx** waiting for its result or a load waiting for noncritical sectors of data and ECC.	PM_CMPLU_STALL_SPEC_FINISH/PM_RUN_INST_CMPL	
ISYNC_STALL_CPI	Completion stall because the ISU is checking the scoreboard for whether the **isync** instruction requires a flush or not.	PM_CMPLU_STALL_ISYNC/PM_RUN_INST_CMPL	
DMISS_LMEM_STALL_CPI	Completion stall due to a cache miss that resolves in local memory.	PM_CMPLU_STALL_DMISS_LMEM/PM_RUN_INST_CMPL	
EXCEPTION_STALL_CPI	Cycles in which the NTC instruction is not allowed to complete because it was interrupted by any exception, which has to be serviced before the instruction can complete.	PM_CMPLU_STALL_EXCEPTION/PM_RUN_INST_CMPL	
NESTED_TEND_STALL_CPI	Completion stall because the ISU is updating the TEXASR to keep track of the nested **tend** and decrement the TEXASR nested level. This is a short delay.	PM_CMPLU_STALL_NESTED_TEND/PM_RUN_INST_CMPL	
VDPLONG_STALL_CPI	Finish stall because the NTF instruction was a scalar multi-cycle instruction issued to the double-precision execution pipe and waiting to finish. Includes binary floating-point instructions in 32- and 64-bit binary floating-point format.	PM_CMPLU_STALL_VDPLONG/PM_RUN_INST_CMPL	
VFXU_STALL_CPI	Finish stall due to a vector fixed-point instruction in the execution pipeline. These instructions are routed to the ALU, ALU2, and DIV pipes.	PM_CMPLU_STALL_VFXU/PM_RUN_INST_CMPL	
LSU_MFSPR_STALL_CPI	Finish stall because the NTF instruction was an **mfspr** instruction targeting an LSU SPR and it is waiting for the register data to be returned.	PM_CMPLU_STALL_LSU_MFSPR/PM_RUN_INST_CMPL	
DARQ_STALL_CPI	Finish stall because the NTF instruction is spending cycles in the DARQ. If this count is large, it is likely because the LSAQ has less than four slots available.	PM_CMPLU_STALL_DARQ/PM_RUN_INST_CMPL	
DPLONG_STALL_CPI	Finish stall because the NTF instruction is a scalar multi-cycle instruction issued to the double-precision execution pipe and waiting to finish. Includes binary floating-point instructions in 32- and 64-bit binary floating point format.	PM_CMPLU_STALL_DPLONG/PM_RUN_INST_CMPL	
STORE_PIPE_ARB_STALL_CPI	Finish stall because the NTF instruction is a store waiting for the next relaunch opportunity after an internal reject. This means the instruction is ready to relaunch and tried once but lost arbitration.	PM_CMPLU_STALL_STORE_PIPE_ARB/PM_RUN_INST_CMPL	

Table 5-23. POWER9 Metric Events and Formulas (Sheet 6 of 23)

Event Name	Event Description	Formula	Unit
ERAT_MISS_STALL_CPI	Finish stall because the NTF instruction is a load or store that suffered a translation miss.	PM_CMPLU_STALL_ERAT_MISS/PM_RUN_INST_CMPL	
LMQ_FULL_STALL_CPI	Finish stall because the NTF instruction is a load that missed in the L1 cache and the LMQ is unable to accept this load miss request because it is full.	PM_CMPLU_STALL_LMQ_FULL/PM_RUN_INST_CMPL	
DMISS_L2L3_CONFLICT_STALL_CPI	Completion stall due to a cache miss that resolves in the L2 or L3 cache with a conflict.	PM_CMPLU_STALL_DMISS_L2L3_CONFLICT/PM_RUN_INST_CMPL	
DMISS_L3MISS_STALL_CPI	Completion stall due to a cache miss resolving missed the L3 cache.	PM_CMPLU_STALL_DMISS_L3MISS/PM_RUN_INST_CMPL	
ST_FWD_STALL_CPI	Completion stall due to store forward	PM_CMPLU_STALL_ST_FWD/PM_RUN_INST_CMPL	
CRYPTO_STALL_CPI	Finish stall because the NTF instruction was routed to the crypto execution pipe and is waiting to finish.	PM_CMPLU_STALL_CRYPTO/PM_RUN_INST_CMPL	
LOAD_FINISH_STALL_CPI	Finish stall because the NTF instruction was a load instruction with all its dependencies satisfied and only going through the LSU pipe to finish.	PM_CMPLU_STALL_LOAD_FINISH/PM_RUN_INST_CMPL	
FXLONG_STALL_CPI	Completion stall due to a long latency scalar fixed-point instruction (division or square root).	PM_CMPLU_STALL_FXLONG/PM_RUN_INST_CMPL	
BRU_STALL_CPI	Completion stall due to a branch unit.	PM_CMPLU_STALL_BRU/PM_RUN_INST_CMPL	
EIEIO_STALL_CPI	Finish stall because the NTF instruction is an EIEIO waiting for a response from the L2 cache.	PM_CMPLU_STALL_EIEIO/PM_RUN_INST_CMPL	
MTFPSCR_STALL_CPI	Completion stall because the ISU is updating the register and notifying the effective address table (EAT).	PM_CMPLU_STALL_MTFPSCR/PM_RUN_INST_CMPL	
LSAQ_ARB_STALL_CPI	Finish stall because the NTF instruction was a load or store that was held in the LSAQ because an older instruction from the SRQ or LRQ won arbitration to the LSU pipe when this instruction tried to launch.	PM_CMPLU_STALL_LSAQ_ARB/PM_RUN_INST_CMPL	
NTC_DISP_FIN_STALL_CPI	Finish stall because the NTF instruction is one that must finish at dispatch.	PM_CMPLU_STALL_NTC_DISP_FIN/PM_RUN_INST_CMPL	
VDP_STALL_CPI	Finish stall because the NTF instruction is a vector instruction issued to the double-precision execution pipe and is waiting to finish.	PM_CMPLU_STALL_VDP/PM_RUN_INST_CMPL	

Table 5-23. POWER9 Metric Events and Formulas (Sheet 7 of 23)

Event Name	Event Description	Formula	Unit
ICT_NOSLOT_DISP_HELD_TBEGIN_CPI	The NTC instruction is held at dispatch because it is a **tbegin** instruction and there is an older **tbegin** in the pipeline that must complete before the younger **tbegin** can dispatch.	PM_ICT_NOSLOT_DISP_HELD_TBEGIN/PM_RUN_INST_CMPL	
NTC_ISSUE_HELD_DARQ_FULL_CPI	The NTC instruction is held at dispatch because there are no slots in the DARQ for it.	PM_NTC_ISSUE_HELD_DARQ_FULL/PM_RUN_INST_CMPL	
ICT_NOSLOT_CYC_CPI	Number of cycles the ICT has no itags assigned to this thread.	PM_ICT_NOSLOT_CYC/PM_RUN_INST_CMPL	
ICT_NOSLOT_IC_MISS_CPI	ICT empty for this thread due to an I-cache miss.	PM_ICT_NOSLOT_IC_MISS/PM_RUN_INST_CMPL	
ICT_NOSLOT_DISP_HELD_ISSQ_CPI	ICT empty for this thread due to a dispatch hold on this thread because: issue queue is full, the BRQ is full, XVCF is full, count cache, Link, Tar is full.	PM_ICT_NOSLOT_DISP_HELD_ISSQ/PM_RUN_INST_CMPL	
NTC_ISSUE_HELD_ARB_CPI	The NTC instruction is held at dispatch because it lost arbitration onto the issue pipe to another instruction (from the same thread or a different thread).	PM_NTC_ISSUE_HELD_ARB/PM_RUN_INST_CMPL	
NTC_FIN_CPI	Cycles in which the oldest instruction in the pipeline (NTC) finishes. This event is used to account for cycles in which work is being completed in the CPI stack.	PM_NTC_FIN/PM_RUN_INST_CMPL	
RUN_CYC_CPI	Run cycles.	PM_RUN_CYC/PM_RUN_INST_CMPL	
ICT_NOSLOT_DISP_HELD_HB_FULL_CPI	ICT empty for this thread due to dispatch holds because the history buffer is full. Can be GPR/VSR/VMR/FPR/CR/XVF.	PM_ICT_NOSLOT_DISP_HELD_HB_FULL/PM_RUN_INST_CMPL	
ICT_NOSLOT_BR_MPRED_ICMISS_CPI	ICT empty for this thread due to an I-cache miss and branch misprediction.	PM_ICT_NOSLOT_BR_MPRED_ICMISS/PM_RUN_INST_CMPL	
NTC_ISSUE_HELD_OTHER_CPI	The NTC instruction is held at dispatch during regular pipeline cycles or because the VSU is busy with multi-cycle instructions, or because of a write-back collision with the VSU.	PM_NTC_ISSUE_HELD_OTHER/PM_RUN_INST_CMPL	
ICT_NOSLOT_IC_L3_CPI	ICT empty for this thread due to I-cache misses that are sourced from the local L3 cache.	PM_ICT_NOSLOT_IC_L3/PM_RUN_INST_CMPL	
ICT_NOSLOT_DISP_HELD_SYNC_CPI	Dispatch held due to a synchronizing instruction at dispatch.	PM_ICT_NOSLOT_DISP_HELD_SYNC/PM_RUN_INST_CMPL	
ICT_NOSLOT_BR_MPRED_CPI	ICT empty for this thread due to a branch misprediction.	PM_ICT_NOSLOT_BR_MPRED/PM_RUN_INST_CMPL	
ICT_NOSLOT_IC_L3MISS_CPI	ICT empty for this thread due to I-cache misses that are sourced from beyond the local L3 cache. The source can be local/remote/distant memory or another core's cache.	PM_ICT_NOSLOT_IC_L3MISS/PM_RUN_INST_CMPL	

Table 5-23. POWER9 Metric Events and Formulas (Sheet 8 of 23)

Event Name	Event Description	Formula	Unit
ICT_NOSLOT_DISP_HELD_CPI	Cycles in which the NTC instruction is held at dispatch for any reason.	PM_ICT_NOSLOT_DISP_HELD/PM_RUN_INST_CMPL	
ICT_NOSLOT_IC_L2_CPI		ICT_NOSLOT_IC_MISS_CPI - ICT_NOSLOT_IC_L3_CPI - ICT_NOSLOT_IC_L3MISS_CPI	
ICT_NOSLOT_DISP_HELD_OTHER_CPI		ICT_NOSLOT_DISP_HELD_CPI - ICT_NOSLOT_DISP_HELD_HB_FULL_CPI - ICT_NOSLOT_DISP_HELD_SYNC_CPI - ICT_NOSLOT_DISP_HELD_TBEGIN_CPI - ICT_NOSLOT_DISP_HELD_ISSQ_CPI	
ISSUE_HOLD_CPI		NTC_ISSUE_HELD_DARQ_FULL_CPI + NTC_ISSUE_HELD_ARB_CPI + NTC_ISSUE_HELD_OTHER_CPI	
SCALAR_STALL_CPI		FXU_STALL_CPI + DP_STALL_CPI + DFU_STALL_CPI + PM_STALL_CPI + CRYPTO_STALL_CPI	
VECTOR_STALL_CPI		VFXU_STALL_CPI + VDP_STALL_CPI	
LSAQ_STALL_CPI		LRQ_FULL_STALL_CPI + SRQ_FULL_STALL_CPI + LSAQ_ARB_STALL_CPI	
EMQ_STALL_CPI		ERAT_MISS_STALL_CPI + EMQ_FULL_STALL_CPI	
LRQ_STALL_CPI		LMQ_FULL_STALL_CPI + ST_FWD_STALL_CPI + LHS_STALL_CPI + LSU_MFSPR_STALL_CPI + LARX_STALL_CPI + LRQ_OTHER_STALL_CPI	
SRQ_STALL_CPI		STORE_DATA_STALL_CPI + EIEIO_STALL_CPI + STCX_STALL_CPI + SLB_STALL_CPI + TEND_STALL_CPI + PASTE_STALL_CPI + TLBIE_STALL_CPI + STORE_PIPE_ARB_STALL_CPI + STORE_FIN_ARB_STALL_CPI	
ICT_NOSLOT_CYC_OTHER_CPI	ICT other stalls.	ICT_NOSLOT_CYC_CPI - ICT_NOSLOT_IC_MISS_CPI - ICT_NOSLOT_BR_MPRED_ICMISS_CPI - ICT_NOSLOT_BR_MPRED_CPI - ICT_NOSLOT_DISP_HELD_CPI	
FXU_OTHER_STALL_CPI	Stalls due to short latency integer operations.	FXU_STALL_CPI - FXLONG_STALL_CPI	
DP_OTHER_STALL_CPI	Stalls due to short latency double-precision operations.	DP_STALL_CPI - DPLONG_STALL_CPI	
DFU_OTHER_STALL_CPI	Stalls due to short latency decimal floating-point operations.	DFU_STALL_CPI - DFLONG_STALL_CPI	
VFXU_OTHER_STALL_CPI	Vector stalls due to small latency integer operations	VFXU_STALL_CPI - VFXLONG_STALL_CPI	
VDP_OTHER_STALL_CPI	Vector stalls due to small latency double-precision operations	VDP_STALL_CPI - VDPLONG_STALL_CPI	
DMISS_L2L3_NOCONFLICT_STALL_CPI	Completion stall due to a cache miss that resolves in the L2 or L3 cache without conflict.	DMISS_L2L3_STALL_CPI - DMISS_L2L3_CONFLICT_STALL_CPI	

Table 5-23. POWER9 Metric Events and Formulas (Sheet 9 of 23)

Event Name	Event Description	Formula	Unit
DMISS_DMEM_STALL_CPI	Completion stall by a D-cache miss that resolved in off-node memory or cache.	DMISS_L3MISS_STALL_CPI - DMISS_L21_L31_STALL_CPI - DMISS_LMEM_STALL_CPI - DMISS_REMOTE_STALL_CPI	
LSU_OTHER_STALL_CPI	Completion LSU stall for other reasons.	LSU_STALL_CPI - LSU_FIN_STALL_CPI - STORE_FINISH_STALL_CPI - SRQ_STALL_CPI - LOAD_FINISH_STALL_CPI - DCACHE_MISS_STALL_CPI - LRQ_STALL_CPI - EMQ_STALL_CPI - LSAQ_STALL_CPI - DARQ_STALL_CPI	
OTHER_STALL_CPI	Completion stall for other reasons.	STALL_CPI - NTC_DISP_FIN_STALL_CPI - NTC_FLUSH_STALL_CPI - ISYNC_STALL_CPI - EXCEPTION_STALL_CPI - LSU_STALL_CPI - EXEC_UNIT_STALL_CPI - BRU_STALL_CPI	
OTHER_CPI	Cycles unaccounted for.	RUN_CPI - NTC_FIN_CPI - THRD_STALL_CPI - STALL_CPI - ISSUE_HOLD_CPI - ICT_NOSLOT_CYC_CPI	
L1_ST_Miss_Ratio(%)	Percentage of L1 store misses per L1 store reference.	PM_ST_MISS_L1 / PM_ST_FIN * 100	%
Flush_Rate(%)	Flush rate.	PM_FLUSH * 100 / PM_RUN_INST_CMPL	%
FXU_All_BUSY	All FXU busy.	PM_FXU_BUSY / PM_CYC	
FXU_All_IDLE	All FXU idle.	PM_FXU_IDLE / PM_CYC	
Loads per inst	PCT instruction loads	PM_LD_REF_L1 / PM_RUN_INST_CMPL	
Stores per inst	PCT instruction stores.	PM_ST_FIN / PM_RUN_INST_CMPL	
Branches_per inst	Branches per instruction.	PM_BRU_FIN / PM_RUN_INST_CMPL	
Fixed per inst	Total fixed-point operations.	PM_FXU_FIN/PM_RUN_INST_CMPL	
dL1_Reload_FROM_L2_Rate(%)	Percentage of DL1 reloads from the L2 cache per instruction.	PM_DATA_FROM_L2 * 100 / PM_RUN_INST_CMPL	%
dL1_Reload_FROM_L2_Miss_Rate(%)	Percentage of DL1 reloads from the L2 cache per instruction.	PM_DATA_FROM_L2MISS * 100 / PM_RUN_INST_CMPL	%
dL1_Reload_FROM_L21_MOD_Rate(%)	Percentage of DL1 reloads from a private L2 cache, other core per instruction.	PM_DATA_FROM_L21_MOD * 100 / PM_RUN_INST_CMPL	%
dL1_Reload_FROM_L21_SHR_Rate(%)	Percentage of DL1 reloads from private L2, other core per Inst.	PM_DATA_FROM_L21_SHR * 100 / PM_RUN_INST_CMPL	%
dL1_Reload_FROM_L31_Rate(%)	Percentage of DL1 reloads from a private L3 cache, other core per instruction.	(PM_DATA_FROM_L31_MOD + PM_DATA_FROM_L31_SHR) * 100 / PM_RUN_INST_CMPL	%
dL1_Reload_FROM_L31_MOD_Rate(%)	Percentage of DL1 reloads from private L3 M state, other core per instruction.	PM_DATA_FROM_L31_MOD * 100 / PM_RUN_INST_CMPL	%
dL1_Reload_FROM_L31_SHR_Rate(%)	Percentage of DL1 reloads from private L3 state, other core per instruction.	PM_DATA_FROM_L31_SHR * 100 / PM_RUN_INST_CMPL	%
dL1_Reload_FROM_RL2L3_MOD_Rate(%)	Percentage of DL1 reloads from private L3, other core per instruction.	PM_DATA_FROM_RL2L3_MOD * 100 / PM_RUN_INST_CMPL	%

Table 5-23. POWER9 Metric Events and Formulas (Sheet 10 of 23)

Event Name	Event Description	Formula	Unit
dL1_Reload_FROM_RL2L3_SHR_Rate(%)	Percentage of DL1 reloads from private L3 cache, other core per instruction.	PM_DATA_FROM_RL2L3_SHR * 100 / PM_RUN_INST_CMPL	%
dL1_Reload_FROM_L3_Rate(%)	Percentage of DL1 reloads from L3 per instruction.	PM_DATA_FROM_L3 * 100 / PM_RUN_INST_CMPL	%
dL1_Reload_FROM_L3_Miss_Rate(%)	Percentage of DL1 reloads from L3 per instruction.	PM_DATA_FROM_L3MISS * 100 / PM_RUN_INST_CMPL	%
dL1_Reload_FROM_DL2L3_MOD_Rate(%)	Percentage of DL1 reloads from Distant L2 or L3 (Modified) per instruction.	PM_DATA_FROM_DL2L3_MOD * 100 / PM_RUN_INST_CMPL	%
dL1_Reload_FROM_DL2L3_SHR_Rate(%)	Percentage of DL1 reloads from Distant L2 or L3 (Shared) per instruction.	PM_DATA_FROM_DL2L3_SHR * 100 / PM_RUN_INST_CMPL	%
dL1_Reload_FROM_LMEM_Rate(%)	Percentage of DL1 reloads from Local Memory per instruction.	PM_DATA_FROM_LMEM * 100 / PM_RUN_INST_CMPL	%
dL1_Reload_FROM_RMEM_Rate(%)	Percentage of DL1 reloads from Remote Memory per instruction.	PM_DATA_FROM_RMEM * 100 / PM_RUN_INST_CMPL	%
dL1_Reload_FROM_DMEM_Rate(%)	Percentage of DL1 reloads from Distant Memory per instruction.	PM_DATA_FROM_DMEM * 100 / PM_RUN_INST_CMPL	%
Taken_Branches(%)	Percentage branches taken.	PM_BR_TAKEN_CMPL * 100 / PM_BRU_FIN	%
BR_Misprediction(%)	Percentage branches mispredicted.	PM_BR_MPRED_CMPL / PM_BR_PRED * 100	%
LSTACK_Mispredict_Rate(%)	Link stack branch misprediction.	PM_BR_MPRED_LSTACK / PM_RUN_INST_CMPL * 100	%
CCACHE_Mispredict_Rate(%)	Count-cache branch misprediction per instruction.	PM_BR_MPRED_CCACHE / PM_RUN_INST_CMPL * 100	%
Br_Mpred_Flush_Rate(%)	Branch mispredict flushes per instruction.	PM_FLUSH_MPRED / PM_RUN_INST_CMPL * 100	%
CCACHE_Misprediction(%)	Count cache branch misprediction.	PM_BR_MPRED_CCACHE / PM_BR_PRED_CCACHE * 100	%
LSTACK_Misprediction(%)	Link stack branch misprediction.	PM_BR_MPRED_LSTACK/ PM_BR_PRED_LSTACK * 100	%
dL1_Miss_Reloads(%)	Percentage of DL1 misses that result in a cache reload.	PM_L1_DCACHE_RELOAD_VALID * 100 / PM_LD_MISS_L1	%
dL1_Reload_FROM_L2(%)	Percentage of DL1 reloads from the L2 cache.	PM_DATA_FROM_L2 * 100 / PM_L1_DCACHE_RELOAD_VALID	%
dL1_Reload_FROM_L2_Miss(%)	Percentage of DL1 reloads that came from the L3 cache and beyond.	PM_DATA_FROM_L2MISS * 100 / PM_L1_DCACHE_RELOAD_VALID	%
L2_Latency	Marked L2 load latency.	PM_MRK_DATA_FROM_L2_CYC/ PM_MRK_DATA_FROM_L2	
L21_MOD_Latency	Marked L21 load latency.	PM_MRK_DATA_FROM_L21_MOD_CYC/ PM_MRK_DATA_FROM_L21_MOD	
L21_SHR_Latency	Marked L21 load latency.	PM_MRK_DATA_FROM_L21_SHR_CYC/ PM_MRK_DATA_FROM_L21_SHR	
L3_Latency	Marked L3 load latency.	PM_MRK_DATA_FROM_L3_CYC/ PM_MRK_DATA_FROM_L3	

Table 5-23. POWER9 Metric Events and Formulas (Sheet 11 of 23)

Event Name	Event Description	Formula	Unit
L31_MOD_Latency	Marked L31 load latency.	PM_MRK_DATA_FROM_L31_MOD_CYC/ PM_MRK_DATA_FROM_L31_MOD	
L31_SHR_Latency	Marked L31 load latency.	PM_MRK_DATA_FROM_L31_SHR_CYC/ PM_MRK_DATA_FROM_L31_SHR	
RL2L3_MOD_Latency	Marked L2/L3 remote load latency.	PM_MRK_DATA_FROM_RL2L3_MOD_CYC/ PM_MRK_DATA_FROM_RL2L3_MOD	
RL2L3_SHR_Latency	Marked L2L3 remote load latency.	PM_MRK_DATA_FROM_RL2L3_SHR_CYC/ PM_MRK_DATA_FROM_RL2L3_SHR	
DL2L3_MOD_Latency	Marked L2L3 remote load latency.	PM_MRK_DATA_FROM_DL2L3_MOD_CYC/ PM_MRK_DATA_FROM_DL2L3_MOD	
DL2L3_SHR_Latency	Marked L2L3 distant load latency.	PM_MRK_DATA_FROM_DL2L3_SHR_CYC/ PM_MRK_DATA_FROM_DL2L3_SHR	
LMEM_Latency	Marked Lmem load latency.	PM_MRK_DATA_FROM_LMEM_CYC/ PM_MRK_DATA_FROM_LMEM	
RMEM_Latency	Marked Rmem load latency.	PM_MRK_DATA_FROM_RMEM_CYC/ PM_MRK_DATA_FROM_RMEM	
DMEM_Latency	Marked Dmem load latency.	PM_MRK_DATA_FROM_DMEM_CYC/ PM_MRK_DATA_FROM_DMEM	
dL1_Reload_FROM_L21_MOD (%)	Percentage of DL1 reloads from the private L2 cache on the other core.	PM_DATA_FROM_L21_MOD * 100 / PM_L1_DCACHE_RELOAD_VALID	%
dL1_Reload_FROM_L21_SHR (%)	Percentage of DL1 reloads from the private L2 cache on the other core.	PM_DATA_FROM_L21_SHR * 100 / PM_L1_DCACHE_RELOAD_VALID	%
dL1_Reload_FROM_L31_MOD (%)	Percentage of DL1 reloads from the private L3 cache on the other core.	PM_DATA_FROM_L31_MOD * 100 / PM_L1_DCACHE_RELOAD_VALID	%
dL1_Reload_FROM_L31_SHR (%)	Percentage of DL1 reloads from the private L3 cache on the other core.	PM_DATA_FROM_L31_SHR * 100 / PM_L1_DCACHE_RELOAD_VALID	%
dL1_Reload_FROM_L3(%)	Percentage of DL1 reloads from the L3 cache.	PM_DATA_FROM_L3 * 100 / PM_L1_DCACHE_RELOAD_VALID	%
dL1_Reload_FROM_L3_Miss(%)	Percentage of DL1 reloads from beyond the local L3 cache.	PM_DATA_FROM_L3MISS * 100 / PM_L1_DCACHE_RELOAD_VALID	%
dL1_Reload_FROM_RL2L3_ SHR(%)	Percentage of DL1 dL1_reloads from a remote L2 or L3 (shared) cache.	PM_DATA_FROM_RL2L3_SHR * 100 / PM_L1_DCACHE_RELOAD_VALID	%
dL1_Reload_FROM_RL2L3_M OD(%)	Percentage of DL1 dL1_Reloads from a remote L2 or L3 (Modified).	PM_DATA_FROM_RL2L3_MOD * 100 / PM_L1_DCACHE_RELOAD_VALID	%
dL1_Reload_FROM_DL2L3_M OD(%)	Percentage of DL1 dL1_Reloads from distant L2 or L3 (Modified)	PM_DATA_FROM_DL2L3_MOD * 100 / PM_L1_DCACHE_RELOAD_VALID	%
dL1_Reload_FROM_DL2L3_S HR(%)	Percentage of DL1 dL1_Reloads from distant L2 or L3 (Shared).	PM_DATA_FROM_DL2L3_SHR * 100 / PM_L1_DCACHE_RELOAD_VALID	%
dL1_Reload_FROM_LMEM(%)	Percentage of DL1 dL1_Reloads from local memory.	PM_DATA_FROM_LMEM * 100 / PM_L1_DCACHE_RELOAD_VALID	%
dL1_Reload_FROM_RMEM(%)	Percentage of DL1 dL1_Reloads from remote memory.	PM_DATA_FROM_RMEM * 100 / PM_L1_DCACHE_RELOAD_VALID	%
dL1_Reload_FROM_DMEM(%)	Percentage of DL1 dL1_Reloads from distant memory.	PM_DATA_FROM_DMEM * 100 / PM_L1_DCACHE_RELOAD_VALID	%

Table 5-23. POWER9 Metric Events and Formulas (Sheet 12 of 23)

Event Name	Event Description	Formula	Unit
MEM_LOCALITY(%)	Memory locality.	PM_DATA_FROM_LMEM * 100/ (PM_DATA_FROM_LMEM + PM_DATA_FROM_RMEM + PM_DATA_FROM_DMEM)	
LD_LMEM_PER_LD_RMEM	Number of loads from local memory per loads from remote memory.	PM_DATA_FROM_LMEM / PM_DATA_FROM_RMEM	
LD_LMEM_PER_LD_DMEM	Number of loads from local memory per loads from distant memory.	PM_DATA_FROM_LMEM / PM_DATA_FROM_DMEM	
LD_LMEM_PER_LD_MEM	Number of loads from local memory per loads from remote and distant memory.	PM_DATA_FROM_LMEM / (PM_DATA_FROM_DMEM + PM_DATA_FROM_RMEM)	
DSLB_Miss_Rate(%)	Percentage DSLB_Miss_Rate per instruction.	PM_DSLB_MISS * 100 / PM_RUN_INST_CMPL	%
DERAT_Miss_Ratio	DERAT miss ratio.	PM_LSU_DERAT_MISS / PM_LSU_DERAT_MISS	
DERAT_4K_Miss_Ratio	DERAT miss ratio for 4 KB page.	PM_DERAT_MISS_4K / PM_LSU_DERAT_MISS	
DERAT_64K_Miss_Ratio	DERAT miss ratio for 64 KB page.	PM_DERAT_MISS_64K / PM_LSU_DERAT_MISS	
DERAT_16M_Miss_Ratio	DERAT miss ratio for 16 MB page (HPT mode).	PM_DERAT_MISS_16M_2M / PM_LSU_DERAT_MISS	
DERAT_16G_Miss_Ratio	DERAT miss ratio for 16 GB page	PM_DERAT_MISS_16G / PM_LSU_DERAT_MISS	
IERAT_Miss_Rate(%)	IERAT miss rate (%).	PM_IERAT_MISS * 100 / PM_RUN_INST_CMPL	%
ISLB_Miss_Rate(%)	Percentage ISLB miss rate per instruction.	PM_ISLB_MISS * 100 / PM_RUN_INST_CMPL	%
DTLB_Miss_Rate(%)	Percentage DTLB miss rate per instruction.	PM_DTLB_MISS / PM_RUN_INST_CMPL *100	%
ITLB_Miss_Rate(%)	Percentage ITLB miss rate per instruction.	PM_ITLB_MISS / PM_RUN_INST_CMPL *100	%
DERAT_MISS_RELOAD(%)	Percentage of DERAT misses that result in an ERAT reload.	PM_DTLB_MISS * 100 / PM_LSU_DERAT_MISS	%
PTEG_FROM_L2(%)	Percentage of DERAT reloads from L2 cache.	PM_DPTEG_FROM_L2 * 100 / PM_DTLB_MISS	%
LD_RMEM_PER_LD_DMEM	Number of loads from remote memory per loads from distant memory	PM_DATA_FROM_RMEM / PM_DATA_FROM_DMEM	
DERAT_4K_Miss_Rate(%)	Percentage DERAT miss rate for 4 KB page per instruction	PM_DERAT_MISS_4K * 100 / PM_RUN_INST_CMPL	%
DERAT_64K_Miss_Rate(%)	Percentage DERAT miss ratio for 64 KB page per instruction	PM_DERAT_MISS_64K * 100 / PM_RUN_INST_CMPL	%
DERAT_16M_Miss_Rate(%)	Percentage DERAT miss rate for 16 MB page per instruction (HPT mode)	PM_DERAT_MISS_16M_2M * 100 / PM_RUN_INST_CMPL	%
DERAT_16G_Miss_Rate(%)	Percentage DERAT miss ratio for 16 GB page per instruction	100 * PM_DERAT_MISS_16G / PM_RUN_INST_CMPL	%
PTEG_FROM_RMEM(%)	Percentage of DERAT reloads from remote memory.	PM_DPTEG_FROM_RMEM * 100 / PM_DTLB_MISS	%

Table 5-23. POWER9 Metric Events and Formulas (Sheet 13 of 23)

Event Name	Event Description	Formula	Unit
PTEG_FROM_DMEM(%)	Percentage of DERAT reloads from distant memory.	PM_DPTEG_FROM_DMEM * 100 / PM_DTLB_MISS	%
PTEG_FROM_L2_Rate(%)	Percentage of DERAT reloads from L2 cache per instruction.	PM_DPTEG_FROM_L2 * 100 / PM_RUN_INST_CMPL	%
PTEG_FROM_L21_MOD_Rate (%)	Percentage of DERAT reloads from private L2 cache, other core per instruction.	PM_DPTEG_FROM_L21_MOD * 100 / PM_RUN_INST_CMPL	%
PTEG_FROM_L21_SHR_Rate (%)	Percentage of DERAT reloads from private L2 cache, other core per instruction.	PM_DPTEG_FROM_L21_SHR * 100 / PM_RUN_INST_CMPL	%
PTEG_FROM_L31_MOD_Rate (%)	Percentage of DERAT reloads from private L3 cache, other core per instruction.	PM_DPTEG_FROM_L31_MOD * 100 / PM_RUN_INST_CMPL	%
PTEG_FROM_L31_SHR_Rate (%)	Percentage of DERAT reloads from private L3 cache, other core per instruction.	PM_DPTEG_FROM_L31_SHR * 100 / PM_RUN_INST_CMPL	%
PTEG_FROM_L3_Rate(%)	Percentage of DERAT reloads from L3 cache per instruction.	PM_DPTEG_FROM_L3 * 100 / PM_RUN_INST_CMPL	%
PTEG_FROM_RL2L3_SHR_ Rate(%)	Percentage of DERAT reloads from remote L2 or L3 (Shared) per instruction.	PM_DPTEG_FROM_RL2L3_SHR * 100 / PM_RUN_INST_CMPL	%
PTEG_FROM_RL2L3_MOD_ Rate(%)	Percentage of DERAT reloads from remote L2 or L3 (Modified) per instruction.	PM_DPTEG_FROM_RL2L3_MOD * 100 / PM_RUN_INST_CMPL	%
PTEG_FROM_L21_MOD(%)	Percentage of DERAT reloads from private L2 cache, other core.	PM_DPTEG_FROM_L21_MOD * 100 / PM_DTLB_MISS	%
PTEG_FROM_L21_SHR(%)	Percentage of DERAT reloads from Private L2, other core	PM_DPTEG_FROM_L21_SHR * 100 / PM_DTLB_MISS	%
PTEG_FROM_L31_MOD(%)	Percentage of DERAT reloads from Private L3, other core.	PM_DPTEG_FROM_L31_MOD * 100 / PM_DTLB_MISS	%
PTEG_FROM_L31_SHR(%)	Percentage of DERAT reloads from Private L3, other core.	PM_DPTEG_FROM_L31_SHR * 100 / PM_DTLB_MISS	%
PTEG_FROM_L3(%)	Percentage of DERAT reloads from L3 cache.	PM_DPTEG_FROM_L3 * 100 / PM_DTLB_MISS	%
PTEG_FROM_RL2L3_SHR(%)	Percentage of DERAT reloads from remote L2 or L3 (Shared).	PM_DPTEG_FROM_RL2L3_SHR * 100 / PM_DTLB_MISS	%
PTEG_FROM_RL2L3_MOD (%)	Percentage of DERAT reloads from remote L2 or L3 (Modified).	PM_DPTEG_FROM_RL2L3_MOD * 100 / PM_DTLB_MISS	%
PTEG_FROM_DL2L3_MOD (%)	Percentage of DERAT reloads from distant L2 or L3 (Modified).	PM_DPTEG_FROM_DL2L3_MOD * 100 / PM_DTLB_MISS	%
PTEG_FROM_DL2L3_SHR(%)	Percentage of DERAT reloads from distant L2 or L3 (Shared)	PM_DPTEG_FROM_DL2L3_SHR * 100 / PM_DTLB_MISS	%
PTEG_FROM_LMEM(%)	Percentage of DERAT reloads from local memory.	PM_DPTEG_FROM_LMEM * 100 / PM_DTLB_MISS	%
PTEG_FROM_DL2L3_MOD_ Rate(%)	Percentage of DERAT reloads from distant L2 or L3 (Modified) per instruction.	PM_DPTEG_FROM_DL2L3_MOD * 100 / PM_RUN_INST_CMPL	%

Table 5-23. POWER9 Metric Events and Formulas (Sheet 14 of 23)

Event Name	Event Description	Formula	Unit
PTEG_FROM_DL2L3_SHR_ Rate(%)	Percentage of DERAT reloads from distant L2 or L3 (Shared) per instruction.	PM_DPTEG_FROM_DL2L3_SHR * 100 / PM_RUN_INST_CMPL	%
PTEG_FROM_LMEM_Rate(%)	Percentage of DERAT reloads from local memory per instruction.	PM_DPTEG_FROM_LMEM * 100 / PM_RUN_INST_CMPL	%
PTEG_FROM_RMEM_Rate(%)	Percentage of DERAT reloads from remote memory per instruction.	PM_DPTEG_FROM_RMEM * 100 / PM_RUN_INST_CMPL	%
PTEG_FROM_DMEM_Rate(%)	Percentage of DERAT reloads from Distant Memory per instruction.	PM_DPTEG_FROM_DMEM * 100 / PM_RUN_INST_CMPL	%
LSU_Reject_Ratio(%)	LSU reject ratio.	PM_LSU_REJECT *100/ (PM_LSU_FIN - PM_LSU_FX_FIN)	%
LHS_Reject_Ratio(%)	LHS reject ratio.	PM_LSU_REJECT_LHS *100/ (PM_LSU_FIN - PM_LSU_FX_FIN)	%
ERAT_Reject_Ratio(%)	ERAT miss reject ratio.	PM_LSU_REJECT_ERAT_MISS * 100 / (PM_LSU_FIN - PM_LSU_FX_FIN)	%
LMQ_full_Reject_Ratio(%)	ERAT miss reject ratio.	PM_LSU_REJECT_LMQ_FULL * 100 / PM_LD_REF_L1	%
Set_Mpred_Reject_Ratio(%)	ERAT miss reject ratio.	PM_LSU_REJECT_SET_MPRED * 100 / (PM_LSU_FIN - PM_LSU_FX_FIN)	%
LSU_Reject_Rate(%)	LSU reject ratio.	PM_LSU_REJECT *100/ PM_RUN_INST_CMPL	%
LHS_Reject_Rate(%)	LHS reject ratio.	PM_LSU_REJECT_LHS *100/ PM_RUN_INST_CMPL	%
ERAT_Reject_Rate(%)	ERAT miss reject ratio.	PM_LSU_REJECT_ERAT_MISS * 100 / PM_RUN_INST_CMPL	%
LMQ_full_Reject_Rate(%)	ERAT miss reject ratio.	PM_LSU_REJECT_LMQ_FULL * 100 / PM_RUN_INST_CMPL	%
Set_Mpred_Reject_Rate(%)	ERAT miss reject ratio.	PM_LSU_REJECT_SET_MPRED * 100 / PM_RUN_INST_CMPL	%
INST_FROM_RL2L3_SHR(%)	Percentage of I-cache reloads from remote L2 or L3 (Shared).	PM_INST_FROM_RL2L3_SHR * 100 / PM_L1_ICACHE_MISS	%
INST_FROM_RL2L3_MOD(%)	Percentage of I-cache reloads from remote L2 or L3 (Modified).	PM_INST_FROM_RL2L3_MOD * 100 / PM_L1_ICACHE_MISS	%
ICache_MISS_RELOAD	I-cache fetches per I-cache miss.	(PM_L1_ICACHE_MISS - PM_IC_PREF_WRITE) / PM_L1_ICACHE_MISS	
INST_FROM_L2(%)	Percentage of I-cache reloads from the L2 cache.	PM_INST_FROM_L2 * 100 / PM_L1_ICACHE_MISS	%
INST_FROM_L21_MOD(%)	Percentage of I-cache reloads from the private L2 cache, other core.	PM_INST_FROM_L21_MOD * 100 / PM_L1_ICACHE_MISS	%
INST_FROM_L21_SHR(%)	Percentage of I-cache reloads from the private L2 cache, other core.	PM_INST_FROM_L21_SHR * 100 / PM_L1_ICACHE_MISS	%
INST_FROM_L3(%)	Percentage of I-cache reloads from the L3 cache.	PM_INST_FROM_L3 * 100 / PM_L1_ICACHE_MISS	%
INST_FROM_DL2L3_MOD(%)	Percentage of I-cache reloads from distant L2 or L3 (Modified).	PM_INST_FROM_DL2L3_MOD * 100 / PM_L1_ICACHE_MISS	%

Table 5-23. POWER9 Metric Events and Formulas (Sheet 15 of 23)

Event Name	Event Description	Formula	Unit
INST_FROM_DL2L3_SHR(%)	Percentage of I-cache reloads from distant L2 or L3 (Shared).	PM_INST_FROM_DL2L3_SHR * 100 / PM_L1_ICACHE_MISS	%
INST_FROM_LMEM(%)	Percentage of I-cache reloads from local memory.	PM_INST_FROM_LMEM * 100 / PM_L1_ICACHE_MISS	%
INST_FROM_RMEM(%)	Percentage of I-cache reloads from remote memory.	PM_INST_FROM_RMEM * 100 / PM_L1_ICACHE_MISS	%
INST_FROM_L31_MOD(%)	Percentage of I-cache reloads from private L3, other core.	PM_INST_FROM_L31_MOD * 100 / PM_L1_ICACHE_MISS	%
INST_FROM_L31_SHR(%)	Percentage of I-cache reloads from private L3, other core.	PM_INST_FROM_L31_SHR * 100 / PM_L1_ICACHE_MISS	%
INST_FROM_DMEM(%)	Percentage of I-cache reloads from distant memory.	PM_INST_FROM_DMEM * 100 / PM_L1_ICACHE_MISS	%
INST_FROM_L2_Rate(%)	Percentage of I-cache reloads from L2 per instruction.	PM_INST_FROM_L2 * 100 / PM_RUN_INST_CMPL	%
INST_FROM_L21_MOD_Rate(%)	Percentage of I-cache reloads from private L2, other core per instruction.	PM_INST_FROM_L21_MOD * 100 / PM_RUN_INST_CMPL	%
INST_FROM_L21_SHR_Rate(%)	Percentage of I-cache reloads from private L2, other core per instruction.	PM_INST_FROM_L21_SHR * 100 / PM_RUN_INST_CMPL	%
INST_FROM_L31_MOD_Rate(%)	Percentage of I-cache reloads from private L3, other core per instruction.	PM_INST_FROM_L31_MOD * 100 / PM_RUN_INST_CMPL	%
INST_FROM_L31_SHR_Rate(%)	Percentage of I-cache reloads from private L3 other core per instruction.	PM_INST_FROM_L31_SHR * 100 / PM_RUN_INST_CMPL	%
INST_FROM_L3_Rate(%)	Percentage of I-cache reloads from L3 cache per instruction.	PM_INST_FROM_L3 * 100 / PM_RUN_INST_CMPL	%
INST_FROM_RL2L3_SHR_Rate(%)	Percentage of I-cache reloads from remote L2 or L3 (Shared) per instruction.	PM_INST_FROM_RL2L3_SHR * 100 / PM_RUN_INST_CMPL	%
INST_FROM_RL2L3_MOD_Rate(%)	Percentage of I-cache reloads from remote L2 or L3 (Modified) per instruction.	PM_INST_FROM_RL2L3_MOD * 100 / PM_RUN_INST_CMPL	%
INST_FROM_DL2L3_MOD_Rate(%)	Percentage of I-cache reloads from distant L2 or L3 (Modified) per instruction.	PM_INST_FROM_DL2L3_MOD * 100 / PM_RUN_INST_CMPL	%
INST_FROM_DL2L3_SHR_Rate(%)	Percentage of I-cache reloads from distant L2 or L3 (Shared) per instruction.	PM_INST_FROM_DL2L3_SHR * 100 / PM_RUN_INST_CMPL	%
INST_FROM_LMEM_Rate(%)	Percentage of I-cache reloads from local memory per instruction.	PM_INST_FROM_LMEM * 100 / PM_RUN_INST_CMPL	%
INST_FROM_RMEM_Rate(%)	Percentage of I-cache reloads from remote memory per instruction.	PM_INST_FROM_RMEM * 100 / PM_RUN_INST_CMPL	%
INST_FROM_DMEM_Rate(%)	Percentage of I-cache reloads from distant memory per instruction	PM_INST_FROM_DMEM * 100 / PM_RUN_INST_CMPL	%
Avg_LMQ_Life_Time	Cycles LMQ slot0 was active on average.	PM_LSU_LMQ_S0_VALID / PM_LSU_LMQ_S0_ALLOC	

Table 5-23. POWER9 Metric Events and Formulas (Sheet 16 of 23)

Event Name	Event Description	Formula	Unit
Avg_LRQ_Life_Time_Even	Average number of cycles LRQ stays active for one load. Slot 0 is valid only for even threads.	PM_LSU_LRQ_S0_VALID / PM_LSU_LRQ_S0_ALLOC	
Avg_LRQ_Life_Time_Odd	Average number of cycles LRQ stays active for one load. Slot 43 is valid only for odd threads.	PM_LSU_LRQ_S43_VALID / PM_LSU_LRQ_S43_ALLOC	
Avg_SRQ_Life_Time_Even	Average number of cycles SRQ stays active for one load. Slot 0 is valid only for even threads.	PM_LSU_SRQ_S0_VALID / PM_LSU_SRQ_S0_ALLOC	
Avg_SRQ_Life_Time_Odd	Average number of cycles SRQ stays active for one load. Slot 39 is valid only for odd threads.	PM_LSU_SRQ_S39_VALID / PM_LSU_SRQ_S39_ALLOC	
Estimated_dL1Miss_Latency	Average L1 miss latency using marked events.	PM_MRK_LD_MISS_L1_CYC / PM_MRK_LD_MISS_L1	
Exposed_dL1Miss_Latency	Estimated exposed miss latency for dL1 misses, such as a load miss when NTC.	PM_MRK_LD_MISS_EXPOSED_CYC / PM_MRK_LD_MISS_EXPOSED_CYC_COUNT	
Custom_secs	Cycles.	PM_RUN_CYC	
L3_Pref_Hit_Ratio(%)	Percentage of L3 prefetch requests that hit in the L3 cache.	PM_L3_PREF_HIT / (PM_L3_PREF_HIT + PM_L3_PREF_MISS) * 100	%
L3_Pref_BW_est	Prefetch bandwidth estimate per L3 cache.	(PM_L3_PREF_MISS * 64) / PM_RUN_CYC	Bytes per PCLK
L3_Pref_BW	Prefetch bandwidth estimate per L3 cache.	(L3_Pref_BW_est * 8 * (proc_freq * 1E-9))	GBps
Mem0_Demand_Read_BW	Bytes/clk.	((PM_MEM0_RQ_DISP - PM_MEM0_PREFETCH_DISP) *8 * 128) / PM_RUN_CYC	Bytes per PCLK
Mem1_Demand_Read_BW	Bytes/clk.	((PM_MEM1_RQ_DISP - PM_MEM1_PREFETCH_DISP) *8 * 128) / PM_RUN_CYC	Bytes per PCLK
Mem0_Speculation	Memory controller dispatches that were speculative.	PM_MEM0_RQ_DISP_SPEC / PM_MEM0_RQ_DISP * 100	
Mem1_Speculation	Memory controller dispatches that were speculative.	PM_MEM1_RQ_DISP_SPEC / PM_MEM1_RQ_DISP * 100	
Mem0_Prefetch_Read_BW	Read GBps.	(PM_MEM0_PREFETCH_DISP *8 * 128) / PM_RUN_CYC	Bytes per PCLK
Mem1_Prefetch_Read_BW	Read GBps.	(PM_MEM1_PREFETCH_DISP *8 * 128) / PM_RUN_CYC	Bytes per PCLK
Mem0_Write_BW	Read GBps.	(PM_MEM0_WQ_DISP *8 * 128) / PM_RUN_CYC	Bytes per PCLK
Mem1_Write_BW	Read GBps.	(PM_MEM1_WQ_DISP *8 * 128) / PM_RUN_CYC	Bytes per PCLK
MC0_Demand_Read_BW	Demand read bandwidth (GBps) MC0.	(Mem0_Demand_Read_BW * (proc_freq * 1E-9))	GBps
MC1_Demand_Read_BW	Demand read bandwidth (GBps) MC1.	(Mem1_Demand_Read_BW * (proc_freq * 1E-9))	GBps
MC0_Prefetch_Read_BW	Prefetch read bandwidth (GBps) MC0.	(Mem0_Prefetch_Read_BW * (proc_freq * 1E-9))	GBps

Table 5-23. POWER9 Metric Events and Formulas (Sheet 17 of 23)

Event Name	Event Description	Formula	Unit
MC1_Prefetch_Read_BW	Prefetch read bandwidth (GBps) MC1.	(Mem1_Prefetch_Read_BW * (proc_freq * 1E-9))	GBps
MC0_Write_BW	Write bandwidth (GBps) MC0.	(Mem0_Write_BW * (proc_freq * 1E-9))	GBps
MC1_Write_BW	Write bandwidth (GBps) MC1.	(Mem1_Write_BW * (proc_freq * 1E-9))	GBps
MC0_Memory_BW	Total memory bandwidth (GBps) MC0.	MC0_Demand_Read_BW + MC0_Prefetch_Read_BW + MC0_Write_BW	GBps
MC1_Memory_BW	Total memory bandwidth (GBps) MC1.	MC1_Demand_Read_BW + MC1_Prefetch_Read_BW + MC1_Write_BW	GBps
Memory_BW	Total memory bandwidth (GBps) MC0 + MC1.	MC0_Memory_BW + MC1_Memory_BW	GBps
L1_Prefetch_Rate(%)	L1 prefetches issued by the prefetch machine per instruction (per thread).	PM_L1_PREF / PM_RUN_INST_CMPL * 100	%
L3_LD_Prefetch_Rate(%)	L3 load prefetches issued by the prefetch machine per instruction (per thread).	PM_L3_PREF_LD / PM_RUN_INST_CMPL * 100	%
L3_ST_Prefetch_Rate(%)	L3 store prefetches issued by the prefetch machine per instruction (per thread).	PM_L3_PREF_ST / PM_RUN_INST_CMPL * 100	%
L3_LDST_Prefetch_Rate(%)	L3 load and store prefetches issued by the prefetch machine per instruction (per thread).	PM_L3_PREF_LDST / PM_RUN_INST_CMPL * 100	%
Prefetch_stream_Alloc_per_ Confirm	Prefetch streams allocated by confirms (per thread).	PM_LSU_DC_PREF_STREAM_ALLOC / PM_LSU_DC_PREF_STREAM_CONF	
Strided_Prefetch_stream_Alloc _per_Confirm	Strided prefetch streams allocated by confirms (per thread).	PM_LSU_DC_PREF_STRIDED_STREAM_ALLOC / PM_LSU_DC_PREF_STRIDED_STREAM_CONF	
L2_LD_Disp(%)	L2 load dispatches as a percentage of total dispatches (per thread).	PM_L2_LD / PM_L2_DISP_ALL * 100	%
L2_ST_Disp(%)	L2 store dispatches as a percentage of total dispatches (per thread)	PM_L2_ST / PM_L2_DISP_ALL * 100	%
L2_INST_Disp(%)	L2 instruction dispatches as a percentage of total dispatches (per thread).	PM_L2_INST / PM_L2_DISP_ALL * 100	%
L2_LD_Miss_Ratio(%)	L2 load misses as a percentage of total L2 load dispatches (per thread).	PM_L2_LD_MISS / PM_L2_LD * 100	%
L2_ld_hit_frequency	Average number of cycles between L2 load hits.	(PM_L2_LD_HIT / PM_RUN_CYC) / 2	
L2_ld_miss_frequency	Average number of cycles between L2 load misses.	(PM_L2_LD_MISS / PM_RUN_CYC) / 2	
L2_ST_Miss_Ratio(%)	L2 store misses as a percentage of total L2 store dispatches (per thread).	PM_L2_ST_MISS / PM_L2_ST * 100	%
L2_INST_Miss_Ratio(%)	L2 instruction misses as a percentage of total L2 instruction dispatches (per thread).	PM_L2_INST_MISS / PM_L2_INST * 100	%

Table 5-23. POWER9 Metric Events and Formulas (Sheet 18 of 23)

Event Name	Event Description	Formula	Unit
L2_Node_Pumps(%)	L2 node pumps as a percentage of all L2 pumps.	(PM_L2_NODE_PUMP) / (PM_L2_NODE_PUMP + PM_L2_SYS_PUMP) * 100	%
L2_Sys_Pumps(%)	L2 system pumps per core as a percentage of all L2 pumps.	(PM_L2_SYS_PUMP) / (PM_L2_NODE_PUMP + PM_L2_SYS_PUMP) * 100	%
L2_IC_Inv_Rate(%)	L2 I-cache invalidates per run instruction (per core).	(PM_L2_IC_INV / 2) / PM_RUN_INST_CMPL * 100	%
L2_DC_Inv_Rate(%)	L2 D-cache invalidates per run instruction (per core).	(PM_L2_DC_INV / 2) / PM_RUN_INST_CMPL * 100	%
L2_Dem_LD_Disp(%)	Demand load misses as a percentage of L2 load dispatches (per thread).	PM_L1_DCACHE_RELOAD_VALID / (PM_L2_LD / 2) * 100	%
L2_Mod_CO(%)	L2 COs that were in M, Me, Mu state as a percentage of all L2 COs.	PM_L2_CASTOUT_MOD / (PM_L2_CASTOUT_MOD + PM_L2_CASTOUT_SHR) * 100	%
L2_Shr_CO(%)	L2 COs that were in T, Te, Si, S state as a percentage of all L2 COs.	PM_L2_CASTOUT_SHR / (PM_L2_CASTOUT_MOD + PM_L2_CASTOUT_SHR) * 100	%
L2_Global_Pred_Correct(%)	L2 global pump prediction success.	PM_L2_GLOB_GUESS_CORRECT / (PM_L2_GLOB_GUESS_CORRECT + PM_L2_GLOB_GUESS_WRONG) * 100	%
L2_Local_Pred_Correct(%)	L2 local pump prediction success.	PM_L2_LOC_GUESS_CORRECT / (PM_L2_LOC_GUESS_CORRECT + PM_L2_LOC_GUESS_WRONG) * 100	%
L2_RC_LD_Disp_Fail(%)	Percentage of L2 load RC dispatch attempts that failed.	(PM_L2_RCLD_DISP_FAIL_ADDR + PM_L2_RCLD_DISP_FAIL_OTHER)/ (PM_L2_RCLD_DISP) * 100	%
L2_RC_LD_Disp_Addr_Fail(%)	Percentage of L2 load RC dispatch attempts that failed because of address collisions and C-class conflicts.	(PM_L2_RCLD_DISP_FAIL_ADDR)/ (PM_L2_RCLD_DISP) * 100	%
L2_RC_ST_Disp_Fail(%)	Percentage of L2 store RC dispatch attempts that failed.	(PM_L2_RCST_DISP_FAIL_ADDR + PM_L2_RCST_DISP_FAIL_OTHER)/ (PM_L2_RCST_DISP) * 100	%
L2_RC_ST_Disp_Addr_Fail(%)	Percentage of L2 store RC dispatch attempts that failed because of address collisions and C-class conflicts.	(PM_L2_RCST_DISP_FAIL_ADDR)/ (PM_L2_RCST_DISP) * 100	%
RC_LD_Busy(%)	Percentage of cycles L2 activated to the core busy for loads due to RC machines being full.	(PM_L2_RCLD_BUSY_RC_FULL)/ (PM_RUN_CYC) * 100	%
RC_ST_Busy(%)	Percentage cycles L2 activated to the core busy for stores due to RC machines being full.	(PM_L2_RCST_BUSY_RC_FULL)/ (PM_RUN_CYC) * 100	%
L2_LD_Rd_Util	Percentage L2 load dispatch attempts cache read utilization (4 pclks per dispatch attempt).	((PM_L2_RCLD_DISP/2)*4)/ (PM_RUN_CYC) * 100	%
L2_ST_Rd_Util	Percentage L2 store dispatch attempts cache read utilization (4 pclks per dispatch attempt).	((PM_L2_RCST_DISP/2)*4)/ (PM_RUN_CYC) * 100	%

Table 5-23. POWER9 Metric Events and Formulas (Sheet 19 of 23)

Event Name	Event Description	Formula	Unit
L2_CO_M_Rd_Util	Percentage of L2 modified CO cache read utilization (4 pclks per dispatch attempt).	((PM_L2_CASTOUT_MOD/2)*4)/ (PM_RUN_CYC) * 100	%
L2_Rd_Util(%)	L2 cache read utilization (per core).	L2_LD_Rd_Util + L2_ST_Rd_Util + L2_CO_M_Rd_Util	%
L2_LDMISS_Wr_Util	L2 load misses that require a cache write (4 pclks per dispatch attempt) percentage of pclks.	(((PM_L2_LD_DISP - PM_L2_LD_HIT)/2)*4)/ (PM_RUN_CYC) * 100	%
L2_ST_Wr_Util	L2 stores that require a cache write (4 pclks per dispatch attempt) percentage of pclks.	((PM_L2_ST_DISP/2)*4)/ (PM_RUN_CYC) * 100	%
L2_Wr_Util(%)	L2 cache write utilization (per core).	L2_LDMISS_Wr_Util + L2_ST_Wr_Util	%
Average_iL1_Miss_Latency	Average I-cache miss latency.	(PM_IC_DEMAND_CYC / PM_IC_DEMAND_REQ)	
dcache_miss_cpi(%)	Data L1 miss portion of CPI.	(LSU_STALL_DCACHE_MISS_CPI/ RUN_CPI) * 100	
L2_cpi(%)	Estimate of data L2 miss rates with measured L2 latency as a percentage of D-cache miss CPI.	(((PM_DATA_FROM_L2 / PM_RUN_INST_CMPL) * L2_Latency) / LSU_STALL_DCACHE_MISS_CPI) *100	%
L3_cpi(%)	Estimate of data L3 miss rates with measured L3 latency as a percentage of D-cache miss CPI.	(((PM_DATA_FROM_L3 / PM_RUN_INST_CMPL) * L3_Latency) / LSU_STALL_DCACHE_MISS_CPI) * 100	%
L21_MOD_cpi(%)	Estimate of data L21 MOD miss rates with measured L21 MOD latency as a percentage of D-cache miss CPI.	(((PM_DATA_FROM_L21_MOD / PM_RUN_INST_CMPL) * L21_MOD_Latency) / LSU_STALL_DCACHE_MISS_CPI) *100	%
L21_SHR_cpi(%)	Estimate of data L21 SHR miss rates with measured L21 SHR latency as a percentage of D-cache miss CPI.	(((PM_DATA_FROM_L21_SHR / PM_RUN_INST_CMPL) * L21_SHR_Latency) / LSU_STALL_DCACHE_MISS_CPI) *100	%
L31_MOD_cpi(%)	Estimate of data L31 MOD miss rates with measured L31 MOD latency as a percentage of D-cache miss CPI.	(((PM_DATA_FROM_L31_MOD / PM_RUN_INST_CMPL) * L31_MOD_Latency) / LSU_STALL_DCACHE_MISS_CPI) *100	%
L31_SHR_cpi(%)	Estimate of data L31 SHR miss rates with measured L31 SHR latency as a percentage of D-cache miss CPI.	(((PM_DATA_FROM_L31_SHR / PM_RUN_INST_CMPL) * L31_SHR_Latency) / LSU_STALL_DCACHE_MISS_CPI) *100	%
RL2L3_MOD_cpi(%)	Estimate of data L2L3 remote MOD miss rates with measured RL2L3 MOD latency as a percentage of D-cache miss CPI.	(((PM_DATA_FROM_RL2L3_MOD / PM_RUN_INST_CMPL) * RL2L3_MOD_Latency) / LSU_STALL_DCACHE_MISS_CPI) *100	%
RL2L3_SHR_cpi(%)	Estimate of data L2L3 shared miss rates with measured RL2L3 SHR latency as a percentage of D-cache miss CPI.	(((PM_DATA_FROM_RL2L3_SHR / PM_RUN_INST_CMPL) * RL2L3_SHR_Latency) / LSU_STALL_DCACHE_MISS_CPI) * 100	%
DL2L3_MOD_cpi(%)	Estimate of data L2L3 distant MOD miss rates with measured DL2L3 MOD latency as a percentage of D-cache miss CPI.	(((PM_DATA_FROM_DL2L3_MOD / PM_RUN_INST_CMPL) * DL2L3_MOD_Latency) / LSU_STALL_DCACHE_MISS_CPI) *100	%

Table 5-23. POWER9 Metric Events and Formulas (Sheet 20 of 23)

Event Name	Event Description	Formula	Unit
DL2L3_SHR_cpi(%)	Estimate of data L2L3 distant SHR miss rates with measured DL2L3 SHR latency as a percentage of D-cache miss CPI.	(((PM_DATA_FROM_DL2L3_SHR / PM_RUN_INST_CMPL) * DL2L3_SHR_Latency) / LSU_STALL_DCACHE_MISS_CPI) *100	%
LMEM_cpi(%)	Estimate of local memory miss rates with measured LMEM latency as a percentage of D-cache miss CPI.	(((PM_DATA_FROM_LMEM / PM_RUN_INST_CMPL) * LMEM_Latency) / LSU_STALL_DCACHE_MISS_CPI) *100	%
RMEM_cpi(%)	Estimate of remote memory miss rates with measured RMEM latency as a percentage of D-cache miss CPI.	(((PM_DATA_FROM_RMEM / PM_RUN_INST_CMPL) * RMEM_Latency) / LSU_STALL_DCACHE_MISS_CPI) *100	%
DMEM_cpi(%)	Estimate of distant memory miss rates with measured DMEM latency as a percentage of D-cache miss CPI.	(((PM_DATA_FROM_DMEM / PM_RUN_INST_CMPL) * DMEM_Latency) / LSU_STALL_DCACHE_MISS_CPI) *100	%
GCT_Empty(%)	GCT empty cycles.	(PM_GCT_NOSLOT_CYC / PM_RUN_CYC) * 100	%
Disp_Flush_Rate(%)	GCT empty cycles.	(PM_FLUSH_DISP / PM_RUN_INST_CMPL) * 100	%
DPTEG_FROM_L2_Rate(%)	Rate of DERAT reloads from the L2 cache.	PM_DPTEG_FROM_L2 * 100 / PM_RUN_INST_CMPL	%
DPTEG_FROM_L3_Rate(%)	Rate of DERAT reloads from the L3 cache.	PM_DPTEG_FROM_L3 * 100 / PM_RUN_INST_CMPL	%
IPTEG_FROM_L2_Rate(%)	Rate of IERAT reloads from the L2 cache.	PM_IPTEG_FROM_L2 * 100 / PM_RUN_INST_CMPL	%
IPTEG_FROM_L3_Rate(%)	Rate of IERAT reloads from the L3 cache.	PM_IPTEG_FROM_L3 * 100 / PM_RUN_INST_CMPL	%
IPTEG_FROM_LMEM_Rate (%)	Rate of IERAT reloads from local memory.	PM_IPTEG_FROM_LMEM * 100 / PM_RUN_INST_CMPL	%
L2_RC_ST_Disp_Fail_Rate(%)	Rate of L2 store dispatches that failed per core.	100 * ((PM_L2_RCST_DISP_FAIL_ADDR + PM_L2_RCST_DISP_FAIL_OTHER)/2) / PM_RUN_INST_CMPL	%
L2_ST_Disp_Rate(%)	Rate of L2 dispatches per core.	100 * (PM_L2_RCST_DISP/2) / PM_RUN_INST_CMPL	%
L2_ST_Disp_Fail_Rate(%)	Rate of L2 store dispatches that failed per core.	100 * ((PM_L2_RCST_DISP_FAIL_ADDR + PM_L2_RCST_DISP_FAIL_OTHER)/2) / PM_RUN_INST_CMPL	%
L31_Latency	Marked L31 load latency.	(PM_MRK_DATA_FROM_L31_SHR_CYC + PM_MRK_DATA_FROM_L31_MOD_CYC) / (PM_MRK_DATA_FROM_L31_SHR + PM_MRK_DATA_FROM_L31_MOD)	
Store_Forward_Rate(%)	Store forward rate.	100 * (PM_LSU0_SRQ_STFWD + PM_LSU1_SRQ_STFWD) / PM_RUN_INST_CMPL	%
Store_Forward_Ratio(%)	Store forward rate.	100 * (PM_LSU0_SRQ_STFWD + PM_LSU1_SRQ_STFWD) / (PM_LD_REF_L1 - PM_LSU_REJECT - PM_LD_MISS_L1)	%
L2_RC_Usage	Average number of RC machines used. One of 16 RC machines is sampled every L2 cycle.	(PM_RC_USAGE / PM_RUN_CYC) * 16	

Table 5-23. POWER9 Metric Events and Formulas (Sheet 21 of 23)

Event Name	Event Description	Formula	Unit
L2_CO_Usage	Average number of CO machines used. One of 16 CO machines is sampled every L2 cycle.	(PM_CO_USAGE / PM_RUN_CYC) * 16	
L2_SN_Usage	Average number of Snoop machines used. One of 8 SN machines is sampled every L2 cycle.	(PM_SN_USAGE / PM_RUN_CYC) * 8	
L2_RC_Lifetime	Average lifetime for L2 RC machine 0.	PM_RC0_BUSY / PM_RC0_DONE	
L2_CO_Lifetime	Average lifetime for L2 Castout machine 0.	PM_CO0_BUSY / PM_CO0_DONE	
L2_SN_Lifetime	Average lifetime for L2 Snoop machine 0.	PM_SN0_BUSY / PM_SN0_DONE	
L2_ST_commands(%)	Percent of stores out of all L2 commands.	PM_L2_ST * 100 / (PM_L2_ST + PM_L2_LD + PM_ISIDE_DISP)	%
L2_LD_commands(%)	Percent of loads out of all L2 commands.	PM_L2_LD * 100 / (PM_L2_ST + PM_L2_LD + PM_ISIDE_DISP)	%
L2_Instr_commands(%)	Percent of instruction reads out of all L2 commands.	PM_ISIDE_DISP * 100 / (PM_L2_ST + PM_L2_LD + PM_ISIDE_DISP)	%
L3_RD_Lifetime	Average lifetime for L3 read machine 0.	PM_L3_RD0_BUSY / PM_L3_RD0_DONE	
L3_PF_Lifetime	Average lifetime for L3 prefetch machine 0.	PM_L3_PF0_BUSY / PM_L3_PF0_DONE	
L3_SN_Lifetime	Average lifetime for L3 snoop machine 0.	PM_L3_SN0_BUSY / PM_L3_SN0_DONE	
L3_CO_Lifetime	Average lifetime for L3 castout machine 0.	PM_L3_CO0_BUSY / PM_L3_CO0_DONE	
L3_WI_Lifetime	Average lifetime for L3 write-in machine 0.	PM_L3_WI0_BUSY / PM_L3_WI0_DONE	
L3_WI_Usage	Average number of write-in machines used. One of 8 WI machines is sampled every L3 cycle.	(PM_L3_WI_USAGE / PM_RUN_CYC) * 8	
L3_ld_hit_frequency	Average number of cycles between L3 load hits.	(PM_L3_LD_HIT / PM_RUN_CYC) / 2	
L3_ld_miss_frequency	Average number of cycles between L3 load misses.	(PM_L3_LD_MISS / PM_RUN_CYC) / 2	
dL1_Reload_FROM_LL4_Rate (%)	Percentage of DL1 reloads from local L4 cache per instruction.	PM_DATA_FROM_LL4 * 100 / PM_RUN_INST_CMPL	%
LL4_Latency	Local L4 average load latency.	PM_MRK_DATA_FROM_LL4_CYC/ PM_MRK_DATA_FROM_LL4	
dL1_Reload_FROM_LL4(%)	Percentage of DL1 dL1_Reloads from the local L4 cache.	PM_DATA_FROM_LL4 * 100 / PM_L1_DCACHE_RELOAD_VALID	%
L4_LOCALITY	L4 locality (percentage).	PM_DATA_FROM_LL4 * 100 / (PM_DATA_FROM_LL4 + PM_DATA_FROM_RL4 + PM_DATA_FROM_DL4)	%
LD_LL4_PER_LD_RL4	Ratio of reloads from local L4 cache to remote L4 cache.	PM_DATA_FROM_LL4 / PM_DATA_FROM_RL4	

Table 5-23. POWER9 Metric Events and Formulas (Sheet 22 of 23)

Event Name	Event Description	Formula	Unit
LD_LL4_PER_LD_DMEM	Ratio of reloads from local L4 cache to distant L4 cache.	PM_DATA_FROM_LL4 / PM_DATA_FROM_DL4	
LD_LL4_PER_LD_MEM	Ratio of reloads from local L4 cache to remote and distant L4 cache.	PM_DATA_FROM_LL4 / (PM_DATA_FROM_DL4 + PM_DATA_FROM_RL4)	
PTEG_FROM_LL4(%)	Percentage of DERAT reloads from local L4 cache.	PM_DPTEG_FROM_LL4 * 100 / PM_DTLB_MISS	%
PTEG_FROM_LL4_Rate(%)	Percentage of DERAT reloads from local L4 per instruction.	PM_DPTEG_FROM_LL4 * 100 / PM_RUN_INST_CMPL	%
INST_FROM_LL4(%)	Percentage of I-cache reloads from local L4 cache.	PM_INST_FROM_LL4 * 100 / PM_L1_ICACHE_MISS	%
INST_FROM_LL4_Rate(%)	Percentage of I-cache reloads from local L4 cache per instruction.	PM_INST_FROM_LL4 * 100 / PM_RUN_INST_CMPL	%
IPTEG_FROM_LL4_Rate(%)	Rate of IERAT reloads from local memory.	PM_IPTEG_FROM_LL4 * 100 / PM_RUN_INST_CMPL	%
dL1_Reload_FROM_RL4_Rate (%)	Percentage of DL1 reloads from remote memory per instruction.	PM_DATA_FROM_RL4 * 100 / PM_RUN_INST_CMPL	%
RL4_Latency	Remote L4 average load latency.	PM_MRK_DATA_FROM_RL4_CYC/ PM_MRK_DATA_FROM_RL4	
dL1_Reload_FROM_RL4(%)	Percentage of DL1 dL1_Reloads from remote L4 cache.	PM_DATA_FROM_RL4 * 100 / PM_L1_DCACHE_RELOAD_VALID	%
PTEG_FROM_RL4(%)	Percentage of DERAT reloads from remote L4 cache.	PM_DPTEG_FROM_RL4 * 100 / PM_DTLB_MISS	%
PTEG_FROM_RL4_Rate(%)	Percentage of DERAT reloads from remote L4 cache per instruction.	PM_DPTEG_FROM_RL4 * 100 / PM_RUN_INST_CMPL	%
INST_FROM_RL4(%)	Percentage of I-cache reloads from remote L4 cache.	PM_INST_FROM_RL4 * 100 / PM_L1_ICACHE_MISS	%
INST_FROM_RL4_Rate(%)	Percentage of I-cache reloads from remote L4 per instruction.	PM_INST_FROM_RL4 * 100 / PM_RUN_INST_CMPL	%
RL4_cpi(%)	Estimate of remote L4 miss rates with measured RL4 latency as a percentage of D-cache miss CPI.	(((PM_DATA_FROM_RL4 / PM_RUN_INST_CMPL) * RL4_Latency) / LSU_STALL_DCACHE_MISS_CPI) *100	%
dL1_Reload_FROM_DL4_Rate (%)	Percentage of DL1 reloads from distant L4 cache per instruction.	PM_DATA_FROM_DL4 * 100 / PM_RUN_INST_CMPL	%
DL4_Latency	Distant L4 average load latency.	PM_MRK_DATA_FROM_DL4_CYC/ PM_MRK_DATA_FROM_DL4	
dL1_Reload_FROM_DL4(%)	Percentage of DL1 dL1_Reloads from distant L4 cache.	PM_DATA_FROM_DL4 * 100 / PM_L1_DCACHE_RELOAD_VALID	%
PTEG_FROM_DL4(%)	Percentage of DERAT reloads from distant L4 cache.	PM_DPTEG_FROM_DL4 * 100 / PM_DTLB_MISS	%
PTEG_FROM_DL4_Rate(%)	Percentage of DERAT reloads from distant L4 cache per instruction.	PM_DPTEG_FROM_DL4 * 100 / PM_RUN_INST_CMPL	%

Table 5-23. POWER9 Metric Events and Formulas (Sheet 23 of 23)

Event Name	Event Description	Formula	Unit
INST_FROM_DL4(%)	Percentage of I-cache reloads from distant L4 cache.	PM_INST_FROM_DL4 * 100 / PM_L1_ICACHE_MISS	%
INST_FROM_DL4_Rate(%)	Percentage of I-cache reloads from distant L4 cache per instruction.	PM_INST_FROM_DL4 * 100 / PM_RUN_INST_CMPL	%
DL4_cpi(%)	Estimate of distant L4 miss rates with measured DL4 latency as a percentage of D-cache miss CPI.	(((PM_DATA_FROM_DL4 / PM_RUN_INST_CMPL) * DL4_Latency) / LSU_STALL_DCACHE_MISS_CPI) *100	%

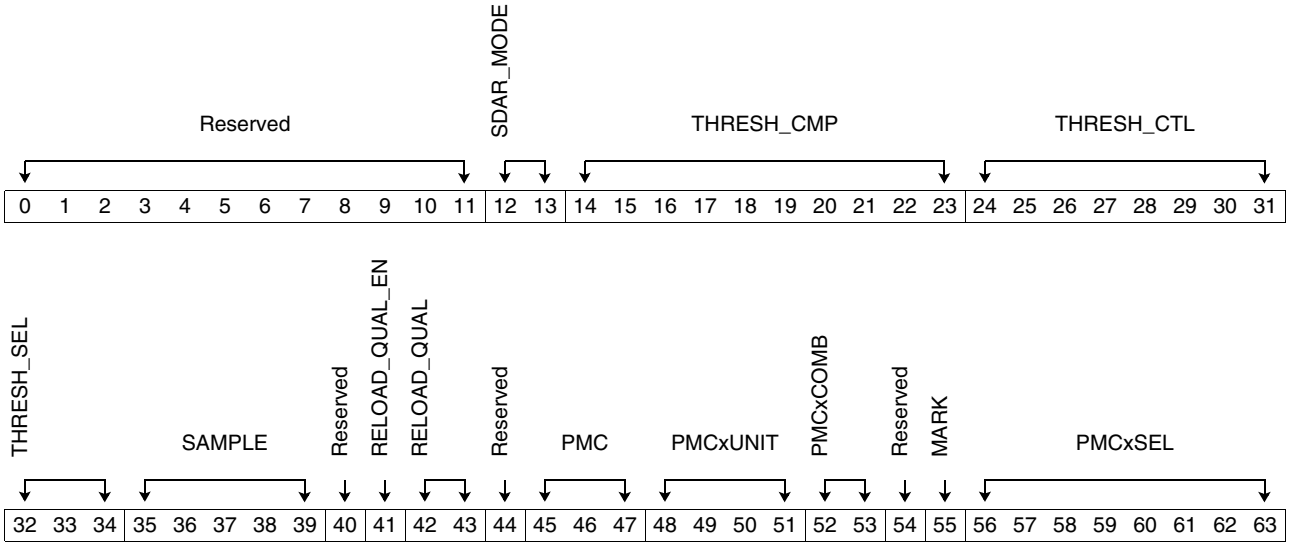
5.15 POWER9 Performance Monitor Event Selection

For each of the four programmable partition-level counters, one event can be selected for monitoring at a given time. The monitored event is selected among the available event sources by setting the corresponding MMCR1[PMCxUNIT] and MMCR1[PMCxSEL] bits to the appropriate values. The quantities counted indicate the number of cycles that the event is active or the number of occurrences of the event, depending on the setting of the MMCR1[PMCxSEL(7)] bit.

5.15.1 Select Event Code Formation

The select event formation code shown in *Figure 5-2* shows how codes from the POWER9 event spreadsheet are translated into MMCR settings.

Figure 5-2. POWER9 Raw Event Coding



Bits	Field Name	MMCR Mapping	MMCR Field
0:11	Reserved		
12:13	SDAR_MODE	MMCRA[20:21] = SDAR_MODE	MMCRA[SDAR_MODE]
14:23	THRESH_CMP	MMCRA[9:11] = THRESH_CMP[0:2] MMCRA[12:18] = THRESH_CMP[3:9]	MMCRA[THRESH_CMP_EXP] MMCRA[THRESH_CMP_MANTISSA]

Bits	Field Name	MMCR Mapping	MMCR Field
24:31	THRESH_CTL	MMCRA[48:51] = THRESH_CTL[0:3] MMCRA[52:55] = THRESH_CTL[4:7]	MMCRA[THRESH_START] MMCRA[THRESH_STOP]
32:34	THRESH_SEL	MMCRA[45:47] = THRESH_SEL	MMCRA[THRESH_EVENT_SEL]
35:39	SAMPLE	MMCRA[57:59] = SAMPLE[0:2] MMCRA[61:62] = SAMPLE[3:4] **Programming Note:** Sampling is not enabled if the bit MARK of the raw event coding is not set. Setting the SAMPLE bit alone is not enough to enable sampling.	MMCRA[RAND_SAMP_ELIG] MMCRA[RAND_SAMP_MODE]
40	Reserved		
41	RELOAD_QUAL_EN	If RELOAD_QUAL_EN = 0, RELOAD_QUAL is ignored. If RELOAD_QUAL_EN = 1, RELOAD_QUAL is used.	
42:43	RELOAD_QUAL	MMCR1[16] = RELOAD_QUAL[0] MMCR1[17] = RELOAD_QUAL[1]	MMCR1[DC_RLD_QUAL] MMCR1[IC_RLD_QUAL]
44	Reserved		
45:47	PMC	If PMC = '000' then software can decide the value of x in PMCxUNIT, PMCxCOMB, PMCxSEL; else if PMC = '001' then PMCxUNIT = PMC1UNIT PMCxCOMB = PMC1COMB PMCxSEL = PMC1SEL; else if PMC = '010' then PMCxUNIT = PMC2UNIT PMCxCOMB = PMC2COMB PMCxSEL = PMC2SEL; else if PMC = '011' then PMCxUNIT = PMC3UNIT PMCxCOMB = PMC3COMB PMCxSEL = PMC3SEL; else if PMC = '100' then PMCxUNIT = PMC4UNIT PMCxCOMB = PMC4COMB PMCxSEL = PMC4SEL	
48:51	PMCxUNIT	If PMC = '001' then MMCR1[0:3] = PMCxUNIT; else if PMC = '010' then MMCR1[4:7] = PMCxUNIT; else if PMC = '011' then MMCR1[8:11] = PMCxUNIT; else if PMC = '100' then MMCR1[12:15] = PMCxUNIT	MMCR1[PMC1UNIT] MMCR1[PMC2UNIT] MMCR1[PMC3UNIT] MMCR1[PMC4UNIT]

Bits	Field Name	MMCR Mapping	MMCR Field
52:53	PMCxCOMB	If PMC = '001' then MMCR1[24:25] = PMCxCOMB; else if PMC = '010' then MMCR1[26:27] = PMCxCOMB; else if PMC = '011' then MMCR1[28:29] = PMCxCOMB; else if PMC = '100' then MMCR1[30:31] = PMCxCOMB	MMCR1[PMC1COMB] MMCR1[PMC2COMB] MMCR1[PMC3COMB] MMCR1[PMC4COMB]
54	Reserved		
55	MARK	MMCRA[63] = MARK	MMCRA[SAMPLE_ENABLE]
56:63	PMCxSEL	If PMC = '001' then MMCR1[32:39] = PMCxSEL; else if PMC = '010' then MMCR1[40:47] = PMCxSEL; else if PMC = '011' then MMCR1[48:55] = PMCxSEL; else if PMC = '100' then MMCR1[56:63] = PMCxSEL	MMCR1[PMC1SEL] MMCR1[PMC2SEL] MMCR1[PMC3SEL] MMCR1[PMC4SEL]

5.16 POWER9 Events Grouping

The POWER9 PMU events are described in previous sections. This section describes some restrictions for grouping events so that the four PMCs can collect four events at a time.

Grouping restrictions are as follows:

- Only one event per PMC
- PMC5 and PMC6 are not programmable
- L2 and L3 events require PMC4 to be programmed with the fourth event from the group.
- MMU events require PMC1 to be programmed for all four threads.

Table 5-24 lists the POWER9 event groups.

Table 5-24. POWER9 Groups (Sheet 1 of 22)

Group ID	Group Name	Group Event Codes	Group Event Names
0	pm_stat	0000010008 0000020002 000003001E 00000400F4	PM_RUN_SPURR PM_INST_CMPL PM_CYC PM_RUN_PURR
1	pm_l1_1	00000100FC 00000200FD 00000300F0 00000400F0	PM_LD_REF_L1 PM_L1_ICACHE_MISS PM_ST_MISS_L1 PM_LD_MISS_L1
2	pm_inst	00000100F2 00000200F2 0000030002 00000400F2	PM_1PLUS_PPC_CMPL PM_INST_DISP PM_INST_CMPL PM_1PLUS_PPC_DISP
3	pm_smt	000001006C 000002006C 000003001E 000004001E	PM_RUN_CYC_ST_MODE PM_RUN_CYC_SMT4_MODE PM_CYC PM_CYC
4	pm_thread	00000100FA 000002000C 00000300F4 000000E8B0	PM_ANY_THRD_RUN_CYC PM_THRD_ALL_RUN_CYC PM_THRD_CONC_RUN_INST PM_TEND_PEND_CYC
5	pm_sync	00000100F0 000002001E 000003001E 0000002884	PM_CYC PM_CYC PM_CYC PM_ISYNC
6	pm_stat	0000002080 00000200F8 000003000C 000004000C	PM_EE_OFF_EXT_INT PM_EXT_INT PM_FREQ_DOWN PM_FREQ_UP
7	pm_l1_2	000001002C 00000200F6 00000300F6 0000000000	PM_L1_DCACHE_RELOADED_ALL PM_LSU_DERAT_MISS PM_L1_DCACHE_RELOAD_VALID PM_SUSPENDED

Table 5-24. POWER9 Groups (Sheet 2 of 22)

Group ID	Group Name	Group Event Codes	Group Event Names
8	pm_flush_1	0000002880 00000050A4 00000058A4 00000400F8	PM_FLUSH_DISP PM_FLUSH_MPRED PM_FLUSH_LSU PM_FLUSH
9	pm_flush_2	0000002084 0000002088 0000030012 0000002888	PM_FLUSH_HB_RESTORE_CYC PM_FLUSH_DISP_SB PM_FLUSH_COMPLETION PM_FLUSH_DISP_TLBIE
10	pm_flush_3	00000020B0 000000C0A8 000000D198 000000D998	PM_LSU_FLUSH_NEXT PM_LSU_FLUSH_CI PM_LSU_FLUSH_ATOMIC PM_LSU_FLUSH_EMSH
11	pm_flush_4	000000D19C 000000D99C 000000D1A0 000000C0B4	PM_LSU_FLUSH_RELAUNCH_MISS PM_LSU_FLUSH_UE PM_LSU_FLUSH_LHS PM_LSU_FLUSH_WRK_ARND
12	pm_flush_5	000000D9A0 000000D1A4 000000D9A4 000000C0BC	PM_LSU_FLUSH_LHL_SHL PM_LSU_FLUSH_SAO PM_LSU_FLUSH_LARX_STCX PM_LSU_FLUSH_OTHER
13	pm_pref_1	000000F084 000002C058 000000F8B0 000000F0B0	PM_PTE_PREFETCH PM_MEM_PREF PM_L3_SW_PREF PM_L3_LD_PREF
14	pm_pref_2	000000E880 0000020054 0000030068 0000004888	PM_L1_SW_PREF PM_L1_PREF PM_L1_ICACHE_RELOADED_PREF PM_IC_PREF_REQ
15	pm_pref_3	000000488C 0000004090 0000004890 0000004094	PM_IC_PREF_WRITE PM_IC_PREF_CANCEL_PAGE PM_IC_PREF_CANCEL_HIT PM_IC_PREF_CANCEL_L2
16	pm_pref_4	000000F0A4 000000F8A4 000000F0A8 000000F0AC	PM_DC_PREF_HW_ALLOC PM_DC_PREF_SW_ALLOC PM_DC_PREF_CONF PM_DC_PREF_STRIDED_CONF
17	pm_inst_mix_1	00000100FE 00000200F0 000000C8BC 000004003E	PM_INST_CMPL PM_ST_CMPL PM_STCX_SUCCESS_CMPL PM_LD_CMPL
18	pm_inst_mix_2	00000100F0 0000024050 0000034054 000004505E	PM_CYC PM_IOPS_CMPL PM_PARTIAL_ST_FIN PM_FLOP_CMPL
19	pm_inst_mix_3	0000010068 000002505C 0000030066 0000040004	PM_BRU_FIN PM_VSU_FIN PM_LSU_FIN PM_FXU_FIN

Table 5-24. POWER9 Groups (Sheet 3 of 22)

Group ID	Group Name	Group Event Codes	Group Event Names
20	pm_inst_mix_4	000001E058 000002E014 000003C058 000004D05E	PM_STCX_FAIL PM_STCX_FIN PM_LARX_FIN PM_BR_CMPL
21	pm_inst_mix_5	000000F8A0 0000024052 000003000E 0000045054	PM_NON_DATA_STORE PM_FXU_IDLE PM_FXU_1PLUS_BUSY PM_FMA_CMPL
22	pm_inst_mix_6	000000F0A0 000002000E 000003005C 000004D04C	PM_DATA_STORE PM_FXU_BUSY PM_BFU_BUSY PM_DFU_BUSY
23	pm_inst_mix_7	00000100FE 0000020016 000003D058 000004D04E	PM_INST_CMPL PM_ST_FIN PM_VSU_DP_FSQRT_FDIV PM_VSU_FSQRT_FDIV
24	pm_cpi_1	000001E054 000002D018 000003003A 000004E018	PM_CMPLU_STALL PM_CMPLU_STALL_EXEC_UNIT PM_CMPLU_STALL_EXCEPTION PM_CMPLU_STALL_NTC_DISP_FIN
25	pm_cpi_2	0000010004 000002C010 0000030004 000004C010	PM_CMPLU_STALL_LRQ_OTHER PM_CMPLU_STALL_LSU PM_CMPLU_STALL_EMQ_FULL PM_CMPLU_STALL_STORE_PIPE_ARB
26	pm_cpi_3	000001002A 000002C014 000003000A 000004C014	PM_CMPLU_STALL_LARX PM_CMPLU_STALL_STORE_FINISH PM_CMPLU_STALL_PM PM_CMPLU_STALL_LMQ_FULL
27	pm_cpi_4	00000100F0 000002C016 0000030014 000004C016	PM_CYC PM_CMPLU_STALL_PASTE PM_CMPLU_STALL_STORE_FIN_ARB PM_CMPLU_STALL_DMISS_L2L3_CONFLICT
28	pm_cpi_5	000001003A 000002C018 0000030016 000004C01A	PM_CMPLU_STALL_LSU_FIN PM_CMPLU_STALL_DMISS_L21_L31 PM_CMPLU_STALL_SRQ_FULL PM_CMPLU_STALL_DMISS_L3MISS
29	pm_cpi_6	000001003C 000002C01A 000003C05C 000004C01C	PM_CMPLU_STALL_DMISS_L2L3 PM_CMPLU_STALL_LHS PM_CMPLU_STALL_VFXU PM_CMPLU_STALL_ST_FWD
30	pm_cpi_7	000001005A 000002C01C 0000030026 000004C01E	PM_CMPLU_STALL_DFLONG PM_CMPLU_STALL_DMISS_REMOTE PM_CMPLU_STALL_STORE_DATA PM_CMPLU_STALL_CRYPTO
31	pm_cpi_8	000001005C 000002D016 0000030038 000004D014	PM_CMPLU_STALL_DP PM_CMPLU_STALL_FXU PM_CMPLU_STALL_DMISS_LMEM PM_CMPLU_STALL_LOAD_FINISH

Table 5-24. POWER9 Groups (Sheet 4 of 22)

Group ID	Group Name	Group Event Codes	Group Event Names
32	pm_cpi_9	000001E050 000002D012 0000030028 000004D016	PM_CMPLU_STALL_TEND PM_CMPLU_STALL_DFU PM_CMPLU_STALL_SPEC_FINISH PM_CMPLU_STALL_FXLONG
33	pm_cpi_10	000001E052 000002D014 000003001E 000004D018	PM_CMPLU_STALL_SLB PM_CMPLU_STALL_LRQ_FULL PM_CYC PM_CMPLU_STALL_BRU
34	pm_cpi_11	000001001C 000002C012 000003C05A 000004C012	PM_CMPLU_STALL_THRD PM_CMPLU_STALL_DCACHE_MISS PM_CMPLU_STALL_VDPLONG PM_CMPLU_STALL_ERAT_MISS
35	pm_cpi_12	000001E05C 000002E018 000003405C 000004D01A	PM_CMPLU_STALL_NESTED_TBEGIN PM_CMPLU_STALL_VFXLONG PM_CMPLU_STALL_DPLONG PM_CMPLU_STALL_EIEIO
36	pm_cpi_13	00000100FC 000002D01C 0000034056 000004E016	PM_LD_REF_L1 PM_CMPLU_STALL_STCX PM_CMPLU_STALL_LSU_MFSPR PM_CMPLU_STALL_LSAQ_ARB
37	pm_cpi_14	0000010006 000002E01E 000003001E 000004405C	PM_DISP_HELD PM_CMPLU_STALL_NTC_FLUSH PM_CYC PM_CMPLU_STALL_VDP
38	pm_cpi_15	0000002880 000002E01C 000003E052 000004E010	PM_FLUSH_DISP PM_CMPLU_STALL_TLBIE PM_ICT_NOSLOT_IC_L3 PM_ICT_NOSLOT_IC_L3MISS
39	pm_cpi_16	00000100F8 000002D01A 0000034058 000004D01E	PM_ICT_NOSLOT_CYC PM_ICT_NOSLOT_IC_MISS PM_ICT_NOSLOT_BR_MPRED_ICMISS PM_ICT_NOSLOT_BR_MPRED
40	pm_cpi_17	0000010064 000002D01E 0000030018 000004E01A	PM_ICT_NOSLOT_DISP_HELD_TBEGIN PM_ICT_NOSLOT_DISP_HELD_ISSQ PM_ICT_NOSLOT_DISP_HELD_HB_FULL PM_ICT_NOSLOT_DISP_HELD
41	pm_cpi_18	000001006A 000002E016 000003D05A 000004D01C	PM_NTC_ISSUE_HELD_DARQ_FULL PM_NTC_ISSUE_HELD_ARB PM_NTC_ISSUE_HELD_OTHER PM_ICT_NOSLOT_DISP_HELD_SYNC
42	pm_cpi_19	00000100F2 000002405A 0000030006 0000040002	PM_1PLUS_PPC_CMPL PM_NTC_FIN PM_CMPLU_STALL_OTHER_CMPL PM_INST_CMPL
43	pm_cpi_20	000001E056 000002E01A 000003003C 000004E012	PM_CMPLU_STALL_FLUSH_ANY_THREAD PM_CMPLU_STALL_LSU_FLUSH_NEXT PM_CMPLU_STALL_NESTED_TEND PM_CMPLU_STALL_MTFPSCR

Table 5-24. POWER9 Groups (Sheet 5 of 22)

Group ID	Group Name	Group Event Codes	Group Event Names
44	pm_data_src_1	00000100FC 000002C04E 000003E054 00000400FE	PM_LD_REF_L1 PM_LD_MISS_L1_FIN PM_LD_MISS_L1 PM_DATA_FROM_MEMORY
45	pm_data_src_2	000001C042 00000200FE 00000300FE 000004C042	PM_DATA_FROM_L2 PM_DATA_FROM_L2MISS PM_DATA_FROM_L3MISS PM_DATA_FROM_L3
46	pm_data_src_3	000001C048 000002C048 000003C04A 000004C04C	PM_DATA_FROM_ON_CHIP_CACHE PM_DATA_FROM_LMEM PM_DATA_FROM_RMEM PM_DATA_FROM_DMEM
47	pm_data_src_4	000001C04C 000002C04A 000003C04C 000004C04A	PM_DATA_FROM_LL4 PM_DATA_FROM_RL4 PM_DATA_FROM_DL4 PM_DATA_FROM_OFF_CHIP_CACHE
48	pm_data_src_5	000001C040 000002C040 000003C040 000004C040	PM_DATA_FROM_L2_NO_CONFLICT PM_DATA_FROM_L2_MEPF PM_DATA_FROM_L2_DISP_CONFLICT_LDHITST PM_DATA_FROM_L2_DISP_CONFLICT_OTHER
49	pm_data_src_6	000001C044 000002C042 000003C042 000004C044	PM_DATA_FROM_L3_NO_CONFLICT PM_DATA_FROM_L3_MEPF PM_DATA_FROM_L3_DISP_CONFLICT PM_DATA_FROM_L31_ECO_MOD
50	pm_data_src_7	000001C04A 000002C046 000003C048 000004C048	PM_DATA_FROM_RL2L3_SHR PM_DATA_FROM_RL2L3_MOD PM_DATA_FROM_DL2L3_SHR PM_DATA_FROM_DL2L3_MOD
51	pm_data_src_8	000001C046 000002C044 000003C044 000004C046	PM_DATA_FROM_L31_SHR PM_DATA_FROM_L31_MOD PM_DATA_FROM_L31_ECO_SHR PM_DATA_FROM_L21_MOD
52	pm_data_src_9	000001C04E 0000020018 000003C046 000004C04E	PM_DATA_FROM_L2MISS_MOD PM_ST_FWD PM_DATA_FROM_L21_SHR PM_DATA_FROM_L3MISS_MOD
53	pm_inst_src_1	0000014042 0000004080 00000300F2 0000044042	PM_INST_FROM_L2 PM_INST_FROM_L1 PM_INST_DISP PM_INST_FROM_L3
54	pm_inst_src_2	000001404E 000002404C 00000300FA 000004404A	PM_INST_FROM_L2MISS PM_INST_FROM_MEMORY PM_INST_FROM_L3MISS PM_INST_FROM_OFF_CHIP_CACHE
55	pm_inst_src_3	000001404C 000002404A 000003404C 0000040012	PM_INST_FROM_LL4 PM_INST_FROM_RL4 PM_INST_FROM_DL4 PM_L1_ICACHE_RELOADED_ALL

Table 5-24. POWER9 Groups (Sheet 6 of 22)

Group ID	Group Name	Group Event Codes	Group Event Names
56	pm_inst_src_4	0000014048 0000024048 000003404A 000004404C	PM_INST_FROM_ON_CHIP_CACHE PM_INST_FROM_LMEM PM_INST_FROM_RMEM PM_INST_FROM_DMEM
57	pm_inst_src_5	0000014046 0000024044 0000034046 0000044046	PM_INST_FROM_L31_SHR PM_INST_FROM_L31_MOD PM_INST_FROM_L21_SHR PM_INST_FROM_L21_MOD
58	pm_inst_src_6	0000014040 0000024040 0000034042 000004404E	PM_INST_FROM_L2_NO_CONFLICT PM_INST_FROM_L2_MEPF PM_INST_FROM_L3_DISP_CONFLICT PM_INST_FROM_L3MISS_MOD
59	pm_inst_src_7	0000014044 0000024042 0000034044 0000044044	PM_INST_FROM_L3_NO_CONFLICT PM_INST_FROM_L3_MEPF PM_INST_FROM_L31_ECO_SHR PM_INST_FROM_L31_ECO_MOD
60	pm_inst_src_8	000001404A 0000024046 0000034048 0000044048	PM_INST_FROM_RL2L3_SHR PM_INST_FROM_RL2L3_MOD PM_INST_FROM_DL2L3_SHR PM_INST_FROM_DL2L3_MOD
61	pm_pteg_src_1	0000015040 0000025040 0000035042 0000045042	PM_IPTEG_FROM_L2_NO_CONFLICT PM_IPTEG_FROM_L2_MEPF PM_IPTEG_FROM_L3_DISP_CONFLICT PM_IPTEG_FROM_L3
62	pm_pteg_src_2	0000015042 0000025042 0000035044 0000045044	PM_IPTEG_FROM_L2 PM_IPTEG_FROM_L3_MEPF PM_IPTEG_FROM_L31_ECO_SHR PM_IPTEG_FROM_L31_ECO_MOD
63	pm_pteg_src_3	0000015044 0000025044 0000035046 0000045046	PM_IPTEG_FROM_L3_NO_CONFLICT PM_IPTEG_FROM_L31_MOD PM_IPTEG_FROM_L21_SHR PM_IPTEG_FROM_L21_MOD
64	pm_pteg_src_4	0000015046 0000025046 0000035048 0000045048	PM_IPTEG_FROM_L31_SHR PM_IPTEG_FROM_RL2L3_MOD PM_IPTEG_FROM_DL2L3_SHR PM_IPTEG_FROM_DL2L3_MOD
65	pm_pteg_src_5	0000015048 0000025048 000003504A 000004504A	PM_IPTEG_FROM_ON_CHIP_CACHE PM_IPTEG_FROM_LMEM PM_IPTEG_FROM_RMEM PM_IPTEG_FROM_OFF_CHIP_CACHE
66	pm_pteg_src_6	000001504A 000002504A 000003504C 000004504C	PM_IPTEG_FROM_RL2L3_SHR PM_IPTEG_FROM_RL4 PM_IPTEG_FROM_DL4 PM_IPTEG_FROM_DMEM
67	pm_pteg_src_7	000001504C 000002504C 0000030002 000004504E	PM_IPTEG_FROM_LL4 PM_IPTEG_FROM_MEMORY PM_INST_CMPL PM_IPTEG_FROM_L3MISS

Table 5-24. POWER9 Groups (Sheet 7 of 22)

Group ID	Group Name	Group Event Codes	Group Event Names
68	pm_pteg_src_8	000001504E 000002F146 0000030002 000004F142	PM_IPTEG_FROM_L2MISS PM_MRK_DPTEG_FROM_RL2L3_MOD PM_INST_CMPL PM_MRK_DPTEG_FROM_L3
69	pm_pteg_src_9	000001E040 000002E040 000003E042 000004E042	PM_DPTEG_FROM_L2_NO_CONFLICT PM_DPTEG_FROM_L2_MEPF PM_DPTEG_FROM_L3_DISP_CONFLICT PM_DPTEG_FROM_L3
70	pm_pteg_src_10	000001E042 000002E042 000003E044 000004E044	PM_DPTEG_FROM_L2 PM_DPTEG_FROM_L3_MEPF PM_DPTEG_FROM_L31_ECO_SHR PM_DPTEG_FROM_L31_ECO_MOD
71	pm_pteg_src_11	000001E044 000002E044 000003E046 000004E046	PM_DPTEG_FROM_L3_NO_CONFLICT PM_DPTEG_FROM_L31_MOD PM_DPTEG_FROM_L21_SHR PM_DPTEG_FROM_L21_MOD
72	pm_pteg_src_12	000001E046 000002E046 000003E048 000004E048	PM_DPTEG_FROM_L31_SHR PM_DPTEG_FROM_RL2L3_MOD PM_DPTEG_FROM_DL2L3_SHR PM_DPTEG_FROM_DL2L3_MOD
73	pm_pteg_src_13	000001E048 000002E048 000003E04A 000004E04A	PM_DPTEG_FROM_ON_CHIP_CACHE PM_DPTEG_FROM_LMEM PM_DPTEG_FROM_RMEM PM_DPTEG_FROM_OFF_CHIP_CACHE
74	pm_pteg_src_14	000001E04A 000002E04A 000003E04C 000004E04C	PM_DPTEG_FROM_RL2L3_SHR PM_DPTEG_FROM_RL4 PM_DPTEG_FROM_DL4 PM_DPTEG_FROM_DMEM
75	pm_pteg_src_15	000001E04C 000002E04C 0000030002 000004E04E	PM_DPTEG_FROM_LL4 PM_DPTEG_FROM_MEMORY PM_INST_CMPL PM_DPTEG_FROM_L3MISS
76	pm_mrk_pteg_1	000001E04E 000002F142 0000030002 000004D058	PM_DPTEG_FROM_L2MISS PM_MRK_DPTEG_FROM_L3_MEPF PM_INST_CMPL PM_VECTOR_FLOP_CMPL
77	pm_mrk_pteg_2	000001F140 0000020002 000003F142 000004F144	PM_MRK_DPTEG_FROM_L2_NO_CONFLICT PM_INST_CMPL PM_MRK_DPTEG_FROM_L3_DISP_CONFLICT PM_MRK_DPTEG_FROM_L31_ECO_MOD
78	pm_mrk_pteg_3	000001F142 000002F14C 000003F144 0000040002	PM_MRK_DPTEG_FROM_L2 PM_MRK_DPTEG_FROM_MEMORY PM_MRK_DPTEG_FROM_L31_ECO_SHR PM_INST_CMPL
79	pm_mrk_pteg_4	000001F144 0000020002 000003F146 000004F14A	PM_MRK_DPTEG_FROM_L3_NO_CONFLICT PM_INST_CMPL PM_MRK_DPTEG_FROM_L21_SHR PM_MRK_DPTEG_FROM_OFF_CHIP_CACHE

Table 5-24. POWER9 Groups (Sheet 8 of 22)

Group ID	Group Name	Group Event Codes	Group Event Names
80	pm_mrk_pteg_5	000001F146 000002F140 0000030002 000004F14C	PM_MRK_DPTEG_FROM_L31_SHR PM_MRK_DPTEG_FROM_L2_MEPF PM_INST_CMPL PM_MRK_DPTEG_FROM_DMEM
81	pm_mrk_pteg_6	000001F148 000002F148 000003F148 0000040002	PM_MRK_DPTEG_FROM_ON_CHIP_CACHE PM_MRK_DPTEG_FROM_LMEM PM_MRK_DPTEG_FROM_DL2L3_SHR PM_INST_CMPL
82	pm_mrk_pteg_7	000001F14C 000002F14A 000003F14C 000004F146	PM_MRK_DPTEG_FROM_LL4 PM_MRK_DPTEG_FROM_RL4 PM_MRK_DPTEG_FROM_DL4 PM_MRK_DPTEG_FROM_L21_MOD
83	pm_mrk_pteg_8	000001F14E 000002F144 0000030002 000004F14E	PM_MRK_DPTEG_FROM_L2MISS PM_MRK_DPTEG_FROM_L31_MOD PM_INST_CMPL PM_MRK_DPTEG_FROM_L3MISS
84	pm_mrk_lat_9	000001D142 000002D14C 000003D14E 000004D12E	PM_MRK_DATA_FROM_L31_ECO_SHR_CYC PM_MRK_DATA_FROM_L31_ECO_SHR PM_MRK_DATA_FROM_DL2L3_MOD PM_MRK_DATA_FROM_DL2L3_MOD_CYC
85	pm_mrk_lat_10	000001415A 000002D148 0000035150 000004C12A	PM_MRK_DATA_FROM_L2_DISP_CONFLICT_LDHITST_CYC PM_MRK_DATA_FROM_L2_DISP_CONFLICT_LDHITST PM_MRK_DATA_FROM_RL2L3_SHR PM_MRK_DATA_FROM_RL2L3_SHR_CYC
86	pm_mrk_lat_11	000001D150 000002C128 000003515C 000004D12A	PM_MRK_DATA_FROM_DL2L3_SHR PM_MRK_DATA_FROM_DL2L3_SHR_CYC PM_MRK_DATA_FROM_RL4 PM_MRK_DATA_FROM_RL4_CYC
87	pm_mrk_lat_12	0000014158 000002C120 000003D14C 000004E11E	PM_MRK_DATA_FROM_L2_NO_CONFLICT_CYC PM_MRK_DATA_FROM_L2_NO_CONFLICT PM_MRK_DATA_FROM_DMEM PM_MRK_DATA_FROM_DMEM_CYC
88	pm_mrk_lat_13	0000014156 000002C126 000003012C 0000040002	PM_MRK_DATA_FROM_L2_CYC PM_MRK_DATA_FROM_L2 PM_MRK_ST_FWD PM_INST_CMPL
89	pm_mrk_lat_14	000001D14C 000002C12E 0000035158 000004D144	PM_MRK_DATA_FROM_LL4 PM_MRK_DATA_FROM_LL4_CYC PM_MRK_DATA_FROM_L31_ECO_MOD_CYC PM_MRK_DATA_FROM_L31_ECO_MOD
90	pm_mrk_lat_15	000001D152 000002C12C 000003D146 000004C124	PM_MRK_DATA_FROM_DL4 PM_MRK_DATA_FROM_DL4_CYC PM_MRK_DATA_FROM_L3_NO_CONFLICT PM_MRK_DATA_FROM_L3_NO_CONFLICT_CYC
91	pm_mrk_lat_16	000001D144 000002C122 000003D144 000004C120	PM_MRK_DATA_FROM_L3_DISP_CONFLICT PM_MRK_DATA_FROM_L3_DISP_CONFLICT_CYC PM_MRK_DATA_FROM_L2_MEPF_CYC PM_MRK_DATA_FROM_L2_MEPF

Table 5-24. POWER9 Groups (Sheet 9 of 22)

Group ID	Group Name	Group Event Codes	Group Event Names
92	pm_mrk_lat_17	000001D146 00000201E0 0000030002 00000401E6	PM_MRK_DATA_FROM_MEMORY_CYC PM_MRK_DATA_FROM_MEMORY PM_INST_CMPL PM_MRK_INST_FROM_L3MISS
93	pm_mrk_lat_18	000001D154 000002D14E 000003D142 000004D128	PM_MRK_DATA_FROM_L21_SHR_CYC PM_MRK_DATA_FROM_L21_SHR PM_MRK_DATA_FROM_LMEM PM_MRK_DATA_FROM_LMEM_CYC
94	pm_mrk_lat_19	000001D148 000002C12A 000003D148 000004D146	PM_MRK_DATA_FROM_RMEM PM_MRK_DATA_FROM_RMEM_CYC PM_MRK_DATA_FROM_L21_MOD_CYC PM_MRK_DATA_FROM_L21_MOD
95	pm_mrk_lat_20	000001415C 000002D142 0000030002 0000040118	PM_MRK_DATA_FROM_L3_MEPF_CYC PM_MRK_DATA_FROM_L3_MEPF PM_INST_CMPL PM_MRK_DCACHE_RELOAD_INTV
96	pm_mrk_lat_21	000001D140 000002D144 0000035156 000004D124	PM_MRK_DATA_FROM_L31_MOD_CYC PM_MRK_DATA_FROM_L31_MOD PM_MRK_DATA_FROM_L31_SHR_CYC PM_MRK_DATA_FROM_L31_SHR
97	pm_mrk_lat_22	000001F14A 0000020002 000003515A 000004D140	PM_MRK_DPTEG_FROM_RL2L3_SHR PM_INST_CMPL PM_MRK_DATA_FROM_ON_CHIP_CACHE_CYC PM_MRK_DATA_FROM_ON_CHIP_CACHE
98	pm_mrk_lat_23	000001D14E 000002D120 0000030002 0000040134	PM_MRK_DATA_FROM_OFF_CHIP_CACHE_CYC PM_MRK_DATA_FROM_OFF_CHIP_CACHE PM_INST_CMPL PM_MRK_INST_TIMEO
99	pm_mrk_lat_24	000001D14A 0000020002 0000035154 000004D142	PM_MRK_DATA_FROM_RL2L3_MOD PM_INST_CMPL PM_MRK_DATA_FROM_L3_CYC PM_MRK_DATA_FROM_L3
100	pm_mrk_lat_25	000001E15E 0000020002 0000035152 00000401E8	PM_MRK_L2_TM_REQ_ABORT PM_INST_CMPL PM_MRK_DATA_FROM_L2MISS_CYC PM_MRK_DATA_FROM_L2MISS
101	pm_mrk_lat_26	000001415E 00000201E4 0000030134 0000040002	PM_MRK_DATA_FROM_L3MISS_CYC PM_MRK_DATA_FROM_L3MISS PM_MRK_ST_CMPL_INT PM_INST_CMPL
102	pm_mrk_1	000001515A 0000020132 0000030132 0000040002	PM_SYNC_MRK_L2MISS PM_MRK_DFU_FIN PM_MRK_VSU_FIN PM_INST_CMPL
103	pm_mrk_2	00000101E4 0000020134 0000030002 0000040132	PM_MRK_L1_ICACHE_MISS PM_MRK_FXU_FIN PM_INST_CMPL PM_MRK_LSU_FIN

Table 5-24. POWER9 Groups (Sheet 10 of 22)

Group ID	Group Name	Group Event Codes	Group Event Names
104	pm_mrk_3	000001016E 000002013A 00000301E4 0000040002	PM_MRK_BR_CMPL PM_MRK_BRU_FIN PM_MRK_BR_MPRED_CMPL PM_INST_CMPL
105	pm_mrk_4	00000101E2 0000024158 000003515E 0000040002	PM_MRK_BR_TAKEN_CMPL PM_MRK_INST PM_MRK_BACK_BR_CMPL PM_INST_CMPL
106	pm_mrk_5	000001013E 000002001E 000003E158 0000040002	PM_MRK_LD_MISS_EXPOSED_CYC PM_CYC PM_MRK_STCX_FAIL PM_INST_CMPL
107	pm_mrk_6	000001F150 0000020130 0000030130 0000040002	PM_MRK_ST_L2DISP_TO_CMPL_CYC PM_MRK_INST_DECODED PM_MRK_INST_FIN PM_INST_CMPL
108	pm_mrk_7	000001F152 0000020002 000003015E 0000040154	PM_MRK_FAB_RSP_BKILL_CYC PM_INST_CMPL PM_MRK_FAB_RSP_CLAIM_RTY PM_MRK_FAB_RSP_BKILL
109	pm_mrk_8	000001D156 00000201E2 0000030162 0000040002	PM_MRK_LD_MISS_L1_CYC PM_MRK_LD_MISS_L1 PM_MRK_LSU_DERAT_MISS PM_INST_CMPL
110	pm_mrk_9	000001D15E 000002001E 0000030002 00000401E0	PM_MRK_RUN_CYC PM_CYC PM_INST_CMPL PM_MRK_INST_CMPL
111	pm_mrk_10	00000100FE 000002F152 0000030154 000004015E	PM_INST_CMPL PM_MRK_FAB_RSP_DCLAIM_CYC PM_MRK_FAB_RSP_DCLAIM PM_MRK_FAB_RSP_RD_RTY
112	pm_mrk_11	00000101EA 0000020138 00000301E2 0000040002	PM_MRK_L1_RELOAD_VALID PM_MRK_ST_NEST PM_MRK_ST_CMPL PM_INST_CMPL
113	pm_mrk_12	0000015154 0000024156 0000030002 0000040116	PM_SYNC_MRK_L3MISS PM_MRK_STCX_FIN PM_INST_CMPL PM_MRK_LARX_FIN
114	pm_mrk_13	00000101E0 000002D15E 000003F150 0000040002	PM_MRK_INST_DISP PM_MRK_DTLB_MISS_16G PM_MRK_ST_DRAIN_TO_L2DISP_CYC PM_INST_CMPL
115	pm_mrk_14	000001015E 000002015E 0000030002 000004F150	PM_MRK_FAB_RSP_RD_T_INTV PM_MRK_FAB_RSP_RWITM_RTY PM_INST_CMPL PM_MRK_FAB_RSP_RWITM_CYC

Table 5-24. POWER9 Groups (Sheet 11 of 22)

Group ID	Group Name	Group Event Codes	Group Event Names
116	pm_mrk_15	00000100F0 000002C124 000003D140 0000040002	PM_CYC PM_MRK_DATA_FROM_L2_DISP_CONFLICT_OTHER PM_MRK_DATA_FROM_L2_DISP_CONFLICT_OTHER_CYC PM_INST_CMPL
117	pm_mrk_16	0000015150 000002001E 000003001E 0000040002	PM_SYNC_MRK_PROBE_NOP PM_CYC PM_CYC PM_INST_CMPL
118	pm_mrk_17	00000100F0 000002001E 000003001E 0000040002	PM_CYC PM_CYC PM_CYC PM_INST_CMPL
119	pm_mrk_18	0000015158 000002D154 000003D154 0000040002	PM_SYNC_MRK_L2HIT PM_MRK_DERAT_MISS_64K PM_MRK_DERAT_MISS_16M PM_INST_CMPL
120	pm_mrk_19	000001D15C 000002D150 000003D152 0000040002	PM_MRK_DTLB_MISS_1G PM_MRK_DERAT_MISS_4K PM_MRK_DERAT_MISS_1G PM_INST_CMPL
121	pm_mrk_20	00000100FE 000002D152 000003D156 00000401E4	PM_INST_CMPL PM_MRK_DERAT_MISS_2M PM_MRK_DTLB_MISS_64K PM_MRK_DTLB_MISS
122	pm_mrk_21	0000015156 000002D156 00000301E6 0000040002	PM_SYNC_MRK_FX_DIVIDE PM_MRK_DTLB_MISS_4K PM_MRK_DERAT_MISS PM_INST_CMPL
123	pm_mrk_22	0000010132 0000020002 000003D15E 000004013A	PM_MRK_INST_ISSUED PM_INST_CMPL PM_MULT_MRK PM_MRK_IC_MISS
124	pm_mrk_23	0000010138 0000020002 000003013E 000004C15E	PM_MRK_BR_2PATH PM_INST_CMPL PM_MRK_STALL_CMPLU_CYC PM_MRK_DTLB_MISS_16M
125	pm_mrk_24	000001F15E 0000020114 0000030002 000004C15C	PM_MRK_PROBE_NOP_CMPL PM_MRK_L2_RC_DISP PM_INST_CMPL PM_MRK_DERAT_MISS_16G
126	pm_mrk_25	00000100F0 000002001E 000003012A 0000040002	PM_CYC PM_CYC PM_MRK_L2_RC_DONE PM_INST_CMPL
127	pm_mrk_26	0000015152 0000020002 000003001E 00000028A4	PM_SYNC_MRK_BR_LINK PM_INST_CMPL PM_CYC PM_MRK_TEND_FAIL

Table 5-24. POWER9 Groups (Sheet 12 of 22)

Group ID	Group Name	Group Event Codes	Group Event Names
128	pm_lsu_1	000000C090 000000C890 0000030002 000004D050	PM_LSU_STCX PM_LSU_NCST PM_INST_CMPL PM_VSU_NON_FLOP_CMPL
129	pm_lsu_4	000000E084 000000E884 000000E088 000000E888	PM_LS0_ERAT_MISS_PREF PM_LS1_ERAT_MISS_PREF PM_LS2_ERAT_MISS_PREF PM_LS3_ERAT_MISS_PREF
130	pm_lsu_5	000000F088 000000F888 000000F08C 000000F88C	PM_LSU0_STORE_REJECT PM_LSU1_STORE_REJECT PM_LSU2_STORE_REJECT PM_LSU3_STORE_REJECT
131	pm_lsu_6	00000100F0 000002001E 000003001E 000004001E	PM_CYC PM_CYC PM_CYC PM_CYC
132	pm_lsu_7	000000C0A0 000000C8A0 000000C0A4 000000C8A4	PM_LSU0_FALSE_LHS PM_LSU1_FALSE_LHS PM_LSU2_FALSE_LHS PM_LSU3_FALSE_LHS
133	pm_lsu_8	00000100F0 000002001E 000003001E 000004001E	PM_CYC PM_CYC PM_CYC PM_CYC
134	pm_lsu_9	000000D088 000000D888 000000D08C 000000D88C	PM_LSU0_LDMX_FIN PM_LSU1_LDMX_FIN PM_LSU2_LDMX_FIN PM_LSU3_LDMX_FIN
135	pm_lsu_10	000000D090 000000D890 000000D094 000000D894	PM_LS0_DC_COLLISIONS PM_LS1_DC_COLLISIONS PM_LS2_DC_COLLISIONS PM_LS3_DC_COLLISIONS
136	pm_lsu_11	000000D0B4 000000D8B4 000000D0BC 000000D8BC	PM_LSU0_SRQ_S0_VALID_CYC PM_LSU0_LRQ_S0_VALID_CYC PM_LSU0_1_LRQF_FULL_CYC PM_LSU2_3_LRQF_FULL_CYC
137	pm_lsu_12	000000E094 000000E894 000000E098 000000E898	PM_LSU0_TM_L1_HIT PM_LSU1_TM_L1_HIT PM_LSU2_TM_L1_HIT PM_LSU3_TM_L1_HIT
138	pm_lsu_13	000000E09C 000000E89C 000000E0A0 000000E8A0	PM_LSU0_TM_L1_MISS PM_LSU1_TM_L1_MISS PM_LSU2_TM_L1_MISS PM_LSU3_TM_L1_MISS
139	pm_lsu_14	000000E0B4 000000E8B4 000000E0B8 000000E8B8	PM_LS0_TM_DISALLOW PM_LS1_TM_DISALLOW PM_LS2_TM_DISALLOW PM_LS3_TM_DISALLOW

Table 5-24. POWER9 Groups (Sheet 13 of 22)

Group ID	Group Name	Group Event Codes	Group Event Names
140	pm_lsu_15	000000F080 000000C8BC 0000030002 000004D05C	PM_LSU_STCX_FAIL PM_STCX_SUCCESS_CMPL PM_INST_CMPL PM_DP_QP_FLOP_CMPL
141	pm_lsu_16	000000F090 000000F890 000000F094 000000F894	PM_LSU0_L1_CAM_CANCEL PM_LSU1_L1_CAM_CANCEL PM_LSU2_L1_CAM_CANCEL PM_LSU3_L1_CAM_CANCEL
142	pm_lsu_17	00000100F0 000002001E 000003001E 000004001E	PM_CYC PM_CYC PM_CYC PM_CYC
143	pm_lsu_18	000001D058 000002E050 000003504E 000004D04A	PM_DARQ0_10_12_ENTRIES PM_DARQ0_7_9_ENTRIES PM_DARQ0_4_6_ENTRIES PM_DARQ0_0_3_ENTRIES
144	pm_lsu_19	000001001A 0000020002 000003E05E 000004505A	PM_LSU_SRQ_FULL_CYC PM_INST_CMPL PM_L3_CO_MEPF PM_SP_FLOP_CMPL
145	pm_lsu_20	000001002E 000002003E 000000408C 000004D056	PM_LMQ_MERGE PM_LSU_LMQ_SRQ_EMPTY_CYC PM_L1_DEMAND_WRITE PM_NON_FMA_FLOP_CMPL
146	pm_lsu_21	0000010062 000002E05E 0000030002 0000040008	PM_LD_L3MISS_PEND_CYC PM_LMQ_EMPTY_CYC PM_INST_CMPL PM_SRQ_EMPTY_CYC
147	pm_lsu_22	000000E080 000000D0AC 000000C09C 000000C89C	PM_S2Q_FULL PM_SRQ_SYNC_CYC PM_LS0_LAUNCH_HELD_PREF PM_LS1_LAUNCH_HELD_PREF
148	pm_lsu_23	00000100F0 000000588C 000000508C 000004505C	PM_CYC PM_SHL_ST_DEP_CREATED PM_SHL_CREATED PM_MATH_FLOP_CMPL
149	pm_lsu_24	000000D0B8 000000D8B8 0000030002 0000045056	PM_LSU_LMQ_FULL_CYC PM_LSU0_LMQ_S0_VALID PM_INST_CMPL PM_SCALAR_FLOP_CMPL
150	pm_l3_1	00000160A0 00000260A0 00000360A0 00000460A0	PM_L3_PF_MISS_L3 PM_L3_CO_MEM PM_L3_PF_ON_CHIP_CACHE PM_L3_PF_ON_CHIP_MEM
151	pm_l3_2	00000168A0 00000268A0 00000368A0 00000468A0	PM_L3_CO_MEPF PM_L3_CO_L31 PM_L3_PF_OFF_CHIP_CACHE PM_L3_PF_OFF_CHIP_MEM

Table 5-24. POWER9 Groups (Sheet 14 of 22)

Group ID	Group Name	Group Event Codes	Group Event Names
152	pm_l3_3	0000010000 00000260A2 00000360A2 00000460A2	PM_SUSPENDED PM_L3_CI_HIT PM_L3_L2_CO_HIT PM_L3_LAT_CI_HIT
153	pm_l3_4	0000010000 00000268A2 00000368A2 00000468A2	PM_SUSPENDED PM_L3_CI_MISS PM_L3_L2_CO_MISS PM_L3_LAT_CI_MISS
154	pm_l3_5	00000160A4 00000260A4 00000360A4 00000460A4	PM_L3_HIT PM_L3_LD_HIT PM_L3_CO_LCO RESERVED
155	pm_l3_6	00000168A4 00000268A4 00000368A4 00000468A4	PM_L3_MISS PM_L3_LD_MISS PM_L3_CINJ PM_L3_TRANS_PF
156	pm_l3_7	00000160A6 00000260A6 00000360A6 00000460A6	PM_TM_SC_CO PM_NON_TM_RST_SC PM_SNP_TM_HIT_M PM_RD_FORMING_SC
157	pm_l3_8	00000168A6 00000268A6 00000368A6 00000468A6	PM_TM_CAM_OVERFLOW PM_TM_RST_SC PM_SNP_TM_HIT_T PM_RD_CLEARING_SC
158	pm_l3_9	0000010000 00000260A8 00000360A8 00000460A8	PM_SUSPENDED PM_L3_PF_HIT_L3 PM_L3_CO PM_SN_HIT
159	pm_l3_10	00000168A8 00000268A8 00000368A8 00000468A8	PM_L3_WI_USAGE PM_RD_HIT_PF PM_SN_INVL PM_SN_MISS
160	pm_l3_11	00000160AA 00000260AA 00000360AA 00000460AA	PM_L3_P0_LCO_NO_DATA PM_L3_P0_LCO_DATA PM_L3_P0_CO_MEM PM_L3_P0_CO_L31
161	pm_l3_12	00000168AA 00000268AA 00000368AA 00000468AA	PM_L3_P1_LCO_NO_DATA PM_L3_P1_LCO_DATA PM_L3_P1_CO_MEM PM_L3_P1_CO_L31
162	pm_l3_13	00000160AC 00000260AC 00000360AC 00000460AC	PM_L3_SN_USAGE PM_L3_PF_USAGE PM_L3_SN0_BUSY PM_L3_SN0_BUSY
163	pm_l3_14	00000168AC 00000268AC 00000368AC 00000468AC	PM_L3_CI_USAGE PM_L3_RD_USAGE PM_L3_CO0_BUSY PM_L3_CO0_BUSY

Table 5-24. POWER9 Groups (Sheet 15 of 22)

Group ID	Group Name	Group Event Codes	Group Event Names
164	pm_l3_15	00000160AE 000002001E 00000360AE 00000460AE	PM_L3_P0_PF_RTY PM_CYC PM_L3_P0_CO_RTY RESERVED
165	pm_l3_16	00000168AE 000002001E 00000368AE 00000468AE	PM_L3_P1_PF_RTY PM_CYC PM_L3_P1_CO_RTY RESERVED
166	pm_l3_17	00000160B0 00000260B0 00000360B0 00000460B0	PM_L3_P0_NODE_PUMP PM_L3_P0_GRP_PUMP PM_L3_P0_SYS_PUMP RESERVED
167	pm_l3_18	00000168B0 00000268B0 00000368B0 00000468B0	PM_L3_P1_NODE_PUMP PM_L3_P1_GRP_PUMP PM_L3_P1_SYS_PUMP RESERVED
168	pm_l3_19	00000160B2 00000260B2 00000360B2 00000460B2	PM_L3_LOC_GUESS_CORRECT PM_L3_SYS_GUESS_CORRECT PM_L3_GRP_GUESS_WRONG_LOW PM_L3_SYS_GUESS_WRONG
169	pm_l3_20	00000168B2 00000268B2 00000368B2 00000468B2	PM_L3_GRP_GUESS_CORRECT PM_L3_LOC_GUESS_WRONG PM_L3_GRP_GUESS_WRONG_HIGH RESERVED
170	pm_l3_21	00000160B4 00000260B4 00000360B4 00000460B4	PM_L3_P0_LCO_RTY PM_L3_P2_LCO_RTY PM_L3_PF0_BUSY PM_L3_PF0_BUSY
171	pm_l3_22	00000168B4 00000268B4 00000368B4 00000468B4	PM_L3_P1_LCO_RTY PM_L3_P3_LCO_RTY PM_L3_RD0_BUSY PM_L3_RD0_BUSY
172	pm_l2_1	0000016080 0000026080 000003609E 0000046080	PM_L2_LD PM_L2_LD_MISS PM_L2_INST PM_L2_DISP_ALL_L2MISS
173	pm_l2_2	0000016880 0000026880 0000036880 0000046880	PM_L2_ST PM_L2_ST_MISS PM_L2_INST_MISS PM_ISIDE_MRU_TOUCH
174	pm_l2_3	0000016082 0000026082 0000036082 0000046082	PM_L2_CASTOUT_MOD PM_L2_IC_INV PM_L2_LD_DISP PM_L2_ST_DISP
175	pm_l2_4	0000016882 0000026882 0000036882 0000046882	PM_L2_CASTOUT_SHR PM_L2_DC_INV PM_L2_LD_HIT PM_L2_ST_HIT

Table 5-24. POWER9 Groups (Sheet 16 of 22)

Group ID	Group Name	Group Event Codes	Group Event Names
176	pm_l2_5	0000016084 0000026084 0000036084 0000046084	PM_L2_RCLD_DISP PM_L2_RCLD_DISP_FAIL_OTHER PM_L2_RCST_DISP PM_L2_RCST_DISP_FAIL_OTHER
177	pm_l2_6	0000016884 0000026884 0000036884 0000046884	PM_L2_RCLD_DISP_FAIL_ADDR PM_DSIDE_MRU_TOUCH PM_L2_RCST_DISP_FAIL_ADDR RESERVED
178	pm_l2_7	0000016086 0000026086 0000036086 0000046086	PM_L2_SN_M_WR_DONE PM_CO_TM_SC_FOOTPRINT PM_L2_RC_ST_DONE PM_L2_SN_M_RD_DONE
179	pm_l2_8	0000016886 0000020000 0000036886 0000046886	PM_CO_DISP_FAIL PM_SUSPENDED PM_L2_SN_SX_I_DONE PM_L2_SN_M_WR_DONE
180	pm_l2_9	0000016088 0000026088 0000036088 0000046088	PM_L2_LOC_GUESS_CORRECT PM_L2_GRP_GUESS_CORRECT PM_L2_SYS_GUESS_CORRECT PM_L2_CHIP_PUMP
181	pm_l2_10	0000016888 0000026888 0000036888 0000046888	PM_L2_LOC_GUESS_WRONG PM_L2_GRP_GUESS_WRONG PM_L2_SYS_GUESS_WRONG PM_L2_GROUP_PUMP
182	pm_l2_11	0000010000 000002608A 000003608A 000004608A	PM_SUSPENDED PM_ISIDE_DISP_FAIL_ADDR PM_L2_RTY_ST RESERVED
183	pm_l2_12	000001688A 000002688A 000003689E 000004688A	PM_ISIDE_DISP PM_ISIDE_DISP_FAIL_OTHER PM_L2_RTY_LD PM_L2_SYS_PUMP
184	pm_l2_13	000001608E 000002608E 000003608E 000004608E	PM_ST_CAUSED_FAIL PM_TM_LD_CONF PM_TM_ST_CONF PM_TM_CAP_OVERFLOW
185	pm_l2_14	000001688E 000002688E 000003688E 000004688E	PM_TM_LD_CAUSED_FAIL PM_TM_FAV_CAUSED_FAIL PM_TM_ST_CAUSED_FAIL RESERVED
186	pm_l2_15	000001608C 000002608C 000003608C 000004608C	PM_RC0_BUSY PM_RC0_BUSY PM_CO0_BUSY PM_CO0_BUSY
187	pm_l2_16	000001688C 000002688C 000003688C 000004688C	PM_RC_USAGE PM_CO_USAGE PM_SN_USAGE RESERVED

Table 5-24. POWER9 Groups (Sheet 17 of 22)

Group ID	Group Name	Group Event Codes	Group Event Names
188	pm_l2_17	0000016090 0000026090 0000030000 0000046090	PM_SN0_BUSY PM_SN0_BUSY PM_SUSPENDED RESERVED
189	pm_l2_18	0000016890 0000026890 0000030000 0000046890	PM_L1PF_L2MEMACC PM_ISIDE_L2MEMACC PM_SUSPENDED RESERVED
190	pm_radix_1	00000100F0 000002001E 000003001E 000004F054	PM_CYC PM_CYC PM_CYC PM_RADIX_PWC_MISS
191	pm_radix_2	000001F058 000002D026 000003F054 000004F056	PM_RADIX_PWC_L2_PTE_FROM_L2 PM_RADIX_PWC_L1_PDE_FROM_L2 PM_RADIX_PWC_L4_PTE_FROM_L3MISS PM_RADIX_PWC_L1_PDE_FROM_L3MISS
192	pm_radix_3	000001F05A 000002D028 000003F058 000004F058	PM_RADIX_PWC_L4_PTE_FROM_L2 PM_RADIX_PWC_L2_PDE_FROM_L2 PM_RADIX_PWC_L1_PDE_FROM_L3 PM_RADIX_PWC_L2_PTE_FROM_L3
193	pm_radix_4	000001F05C 000002D02A 000003F05A 000004F05A	PM_RADIX_PWC_L3_PDE_FROM_L3 PM_RADIX_PWC_L3_PDE_FROM_L2 PM_RADIX_PWC_L2_PDE_FROM_L3 PM_RADIX_PWC_L4_PTE_FROM_L3
194	pm_radix_5	000000F098 000000F898 0000030000 000004F05C	PM_XLATE_HPT_MODE PM_XLATE_RADIX_MODE PM_SUSPENDED PM_RADIX_PWC_L2_PTE_FROM_L3MISS
195	pm_radix_6	000000F898 000002D02E 000003F05E 000004F05E	PM_XLATE_RADIX_MODE PM_RADIX_PWC_L3_PTE_FROM_L2 PM_RADIX_PWC_L3_PTE_FROM_L3 PM_RADIX_PWC_L3_PTE_FROM_L3MISS
196	pm_erat_1	000000E08C 000000E88C 000000E090 000000E890	PM_LSU0_ERAT_HIT PM_LSU1_ERAT_HIT PM_LSU2_ERAT_HIT PM_LSU3_ERAT_HIT
197	pm_tbl_walk	0000010026 000000E0BC 000000E8BC 000004001E	PM_TABLEWALK_CYC PM_LS0_PTE_TABLEWALK_CYC PM_LS1_PTE_TABLEWALK_CYC PM_CYC
198	pm_erat_2	0000000000 0000020064 000003006A 000004C05A	PM_SUSPENDED PM_IERAT_RELOAD_4K PM_IERAT_RELOAD_64K PM_DTLB_MISS_1G
199	pm_erat_3	0000000000 000002C054 000003001E 000004006A	PM_SUSPENDED PM_DERAT_MISS_64K PM_CYC PM_IERAT_RELOAD_16M

Table 5-24. POWER9 Groups (Sheet 18 of 22)

Group ID	Group Name	Group Event Codes	Group Event Names
200	pm_slb	00000100F6 000000D0A8 000000D8A8 000000F89C	PM_IERAT_RELOAD PM_DSLB_MISS PM_ISLB_MISS PM_XLATE_MISS
201	pm_dtlb	0000000000 0000000000 0000000000 0000000000	PM_SUSPENDED PM_SUSPENDED PM_SUSPENDED PM_SUSPENDED
202	pm_tlb_1	000001F054 0000020066 0000030058 00000400FC	PM_TLB_HIT PM_TLB_MISS PM_TLBIE_FIN PM_ITLB_MISS
203	pm_derat	000001C056 000002C056 000003C054 000004C054	PM_DERAT_MISS_4K PM_DTLB_MISS_4K PM_DERAT_MISS_16M_2M PM_DERAT_MISS_16G
204	pm_tlb_2	000000F880 000000F884 00000300FC 00000050A8	PM_SNOOP_TLBIE PM_TABLEWALK_CYC_PREF PM_DTLB_MISS PM_EAT_FORCE_MISPRED
205	pm_br_1	000001515C 00000040AC 00000048AC 00000040B0	PM_SYNC_MRK_BR_MPRED PM_BR_MPRED_CCACHE PM_BR_MPRED_LSTACK PM_BR_PRED_TAKEN_CR
206	pm_br_2	00000048B0 00000200FA 00000040B4 00000040B8	PM_BR_MPRED_PCACHE PM_BR_TAKEN_CMPL PM_BR_PRED_TA PM_BR_MPRED_TAKEN_CR
207	pm_br_3	000000409C 000002505E 00000040A0 000000489C	PM_BR_PRED PM_BACK_BR_CMPL PM_BR_UNCOND PM_BR_CORECT_PRED_TAKEN_CMPL
208	pm_br_4	00000048A0 00000040A4 00000040A8 0000040036	PM_BR_PRED_PCACHE PM_BR_PRED_CCACHE PM_BR_PRED_LSTACK PM_BR_2PATH
209	pm_br_5	00000048B8 0000020056 00000058A0 00000400F6	PM_BR_MPRED_TAKEN_TA PM_TAKEN_BR_MPRED_CMPL PM_LINK_STACK_CORRECT PM_BR_MPRED_CMPL
210	pm_br_6	00000050B0 00000058B0 00000050B4 00000058B4	PM_BTAC_BAD_RESULT PM_BTAC_GOOD_RESULT PM_TAGE_CORRECT_TAKEN_CMPL PM_TAGE_CORRECT
211	pm_br_7	00000050B8 00000058B8 0000005098 0000005898	PM_TAGE_OVERRIDE_WRONG PM_TAGE_OVERRIDE_WRONG_SPEC PM_LINK_STACK_WRONG_ADD_PRED PM_LINK_STACK_INVALID_PTR

Table 5-24. POWER9 Groups (Sheet 19 of 22)

Group ID	Group Name	Group Event Codes	Group Event Names
212	pm_disp	00000028B0 000000208C 0000030008 000000288C	PM_DISP_HELD_TBEGIN PM_CLB_HELD PM_DISP_STARVED PM_DISP_CLB_HELD_BAL
213	pm_tm_1	0000002090 0000002890 0000002094 0000002894	PM_DISP_CLB_HELD_SB PM_DISP_CLB_HELD_TLBIE PM_TM_OUTER_TBEGIN PM_TM_OUTER_TEND
214	pm_tm_	0000002098 0000002898 000000209C 000000289C	PM_TM_NESTED_TEND PM_TM_TABORT_TRECLAIM PM_TM_FAV_TBEGIN PM_TM_NON_FAV_TBEGIN
215	pm_tm_3	00000020A0 00000028A0 00000020A4 00000020A8	PM_TM_NESTED_TBEGIN PM_TM_TSUSPEND PM_TM_TRESUME PM_TM_FAIL_FOOTPRINT_OVERFLOW
216	pm_tm_4	00000028A8 00000020AC 00000028AC 000004D05A	PM_TM_FAIL_CONF_NON_TM PM_TM_FAIL_CONF_TM PM_TM_FAIL_SELF PM_NON_MATH_FLOP_CMPL
217	pm_tm_5	000000E0A4 000000E0AC 000000E8AC 000000E0B0	PM_TMA_REQ_L2 PM_TM_FAIL_TLBIE PM_TM_FAIL_TX_CONFLICT PM_TM_FAIL_NON_TX_CONFLICT
218	pm_tm_6	0000010060 000002E012 0000030060 000004001E	PM_TM_TRANS_RUN_CYC PM_TM_TX_PASS_RUN_CYC PM_TM_TRANS_RUN_INST PM_CYC
219	pm_isu	0000003884 0000003080 0000003880 0000003084	PM_ISU3_ISS_HOLD_ALL PM_ISU0_ISS_HOLD_ALL PM_ISU2_ISS_HOLD_ALL PM_ISU1_ISS_HOLD_ALL
220	pm_dpref	000000F8A8 000002001E 000000F8AC 000004001E	PM_DC_PREF_FUZZY_CONF PM_CYC PM_DC_DEALLOC_NO_CONF PM_CYC
221	pm_thr_prio	0000005880 00000040BC 00000048BC 0000005080	PM_THRD_PRIO_6_7_CYC PM_THRD_PRIO_0_1_CYC PM_THRD_PRIO_2_3_CYC PM_THRD_PRIO_4_5_CYC
222	pm_decode_1	00000058A8 000002001E 000003001E 000004001E	PM_DECODE_HOLD_ICT_FULL PM_CYC PM_CYC PM_CYC
223	pm_decode_2	00000048A4 00000048B4 00000048A8 0000004898	PM_STOP_FETCH_PENDING_CYC PM_DECODE_FUSION_CONST_GEN PM_DECODE_FUSION_LD_ST_DISP PM_IC_DEMAND_L2_BR_REDIRECT

Table 5-24. POWER9 Groups (Sheet 20 of 22)

Group ID	Group Name	Group Event Codes	Group Event Names
224	pm_icache	0000005888 0000005094 0000004894 0000004098	PM_IC_INVALIDATE PM_IC_MISS_ICBI PM_IC_RELOAD_PRIVATE PM_IC_DEMAND_L2_BHT_REDIRECT
225	pm_reject_1	0000004880 000002E05C 000003001C 000004E05C	PM_BANK_CONFLICT PM_LSU_REJECT_ERAT_MISS PM_LSU_REJECT_LMQ_FULL PM_LSU_REJECT_LHS
226	pm_reject_2	0000004084 000002E05A 0000030064 000004405E	PM_EAT_FULL_CYC PM_LRQ_REJECT PM_DARQ_STORE_XMIT PM_DARQ_STORE_REJECT
227	pm_ibuff	0000010018 0000004088 0000004884 000004001C	PM_IC_DEMAND_CYC PM_IC_DEMAND_REQ PM_IBUF_FULL_CYC PM_INST_IMC_MATCH_CMPL
228	pm_mem	0000010056 000002001E 000003C05E 000004C058	PM_MEM_READ PM_CYC PM_MEM_RWITM PM_MEM_CO
229	pm_ict	0000010028 0000020008 000003001A 0000045058	PM_STALL_END_ICT_EMPTY PM_ICT_EMPTY_CYC PM_DATA_TABLEWALK_CYC PM_IC_MISS_CMPL
230	pm_probe	0000010058 000002000A 000003001E 0000040014	PM_MEM_LOC_THRESH_IFU PM_HV_CYC PM_CYC PM_PROBE_NOP_DISP
231	pm_mem	000001C05E 000002001A 000003006E 0000040056	PM_MEM_LOC_THRESH_LSU_MED PM_NTC_ALL_FIN PM_NEST_REF_CLK PM_MEM_LOC_THRESH_LSU_HIGH
232	pm_pump_1	0000010050 0000020050 0000030050 0000040050	PM_CHIP_PUMP_CPRED PM_GRP_PUMP_CPRED PM_SYS_PUMP_CPRED PM_SYS_PUMP_MPRED_RTY
233	pm_pump_2	0000010052 0000020052 0000030052 0000040052	PM_GRP_PUMP_MPRED_RTY PM_GRP_PUMP_MPRED PM_SYS_PUMP_MPRED PM_PUMP_MPRED
234	pm_pump_3	0000010054 000002001E 000003001E 0000045050	PM_PUMP_CPRED PM_CYC PM_CYC PM_1FLOP_CMPL
235	pm_pump_4	000001C050 000002C050 000003C050 000004C050	PM_DATA_CHIP_PUMP_CPRED PM_DATA_GRP_PUMP_CPRED PM_DATA_SYS_PUMP_CPRED PM_DATA_SYS_PUMP_MPRED_RTY

Table 5-24. POWER9 Groups (Sheet 21 of 22)

Group ID	Group Name	Group Event Codes	Group Event Names
236	pm_pump_5	000001C052 000002C052 000003C052 000004C052	PM_DATA_GRP_PUMP_MPRED_RTY PM_DATA_GRP_PUMP_MPRED PM_DATA_SYS_PUMP_MPRED PM_DATA_PUMP_MPRED
237	pm_pump_6	000001C054 000002001E 000003001E 000004D052	PM_DATA_PUMP_CPRED PM_CYC PM_CYC PM_2FLOP_CMPL
238	pm_pump_7	0000014050 000002C05C 0000034050 0000044050	PM_INST_CHIP_PUMP_CPRED PM_INST_GRP_PUMP_CPRED PM_INST_SYS_PUMP_CPRED PM_INST_SYS_PUMP_MPRED_RTY
239	pm_pump_8	0000014052 000002C05E 0000034052 0000044052	PM_INST_GRP_PUMP_MPRED_RTY PM_INST_GRP_PUMP_MPRED PM_INST_SYS_PUMP_MPRED PM_INST_PUMP_MPRED
240	pm_pump_9	0000014054 000002001E 000003001E 0000045052	PM_INST_PUMP_CPRED PM_CYC PM_CYC PM_4FLOP_CMPL
241	pm_flop_1	00000100FE 000002011C 000003F14A 000004D054	PM_INST_CMPL PM_MRK_NTC_CYC PM_MRK_DPTEG_FROM_RMEM PM_8FLOP_CMPL
242	pm_darq_2	00000100FE 0000020058 000003005A 000004000A	PM_INST_CMPL PM_DARQ1_10_12_ENTRIES PM_ISQ_0_8_ENTRIES PM_ISQ_36_44_ENTRIES
243	pm_darq_1	00000100FE 000002005A 000003E050 000004C122	PM_INST_CMPL PM_DARQ1_7_9_ENTRIES PM_DARQ1_4_6_ENTRIES PM_DARQ1_0_3_ENTRIES
244	pm_aso_dsource1	000001D154 000002D144 000003D14C 000004E11E	PM_MRK_DATA_FROM_L21_SHR_CYC PM_MRK_DATA_FROM_L31_MOD PM_MRK_DATA_FROM_DMEM PM_MRK_DATA_FROM_DMEM_CYC
245	pm_aso_dsource2	000001D140 000002D14E 000003D148 000004D146	PM_MRK_DATA_FROM_L31_MOD_CYC PM_MRK_DATA_FROM_L21_SHR PM_MRK_DATA_FROM_L21_MOD_CYC PM_MRK_DATA_FROM_L21_MOD
246	pm_aso_dsource3	000001D14A 000002C128 0000035150 000004C12A	PM_MRK_DATA_FROM_RL2L3_MOD PM_MRK_DATA_FROM_DL2L3_SHR_CYC PM_MRK_DATA_FROM_RL2L3_SHR PM_MRK_DATA_FROM_RL2L3_SHR_CYC
247	pm_aso_dsource4	000001D150 000002D14A 000003C04A 000004D142	PM_MRK_DATA_FROM_DL2L3_SHR PM_MRK_DATA_FROM_RL2L3_MOD_CYC PM_DATA_FROM_RMEM PM_MRK_DATA_FROM_L3

Table 5-24. POWER9 Groups (Sheet 22 of 22)

Group ID	Group Name	Group Event Codes	Group Event Names
248	pm_aso_dsource5	0000014156 000002C126 0000035156 000004D124	PM_MRK_DATA_FROM_L2_CYC PM_MRK_DATA_FROM_L2 PM_MRK_DATA_FROM_L31_SHR_CYC PM_MRK_DATA_FROM_L31_SHR
249	pm_aso_dsource6	000001D148 000002C12A 000003D14E 000004D12E	PM_MRK_DATA_FROM_RMEM PM_MRK_DATA_FROM_RMEM_CYC PM_MRK_DATA_FROM_DL2L3_MOD PM_MRK_DATA_FROM_DL2L3_MOD_CYC
250	pm_aso_dsource7	000001D14A 000002C048 000003D142 000004D128	PM_MRK_DATA_FROM_RL2L3_MOD PM_DATA_FROM_LMEM PM_MRK_DATA_FROM_LMEM PM_MRK_DATA_FROM_LMEM_CYC
251	pm_aso_dsource8	000001D150 000002D14A 0000035154 000004D142	PM_MRK_DATA_FROM_DL2L3_SHR PM_MRK_DATA_FROM_RL2L3_MOD_CYC PM_MRK_DATA_FROM_L3_CYC PM_MRK_DATA_FROM_L3

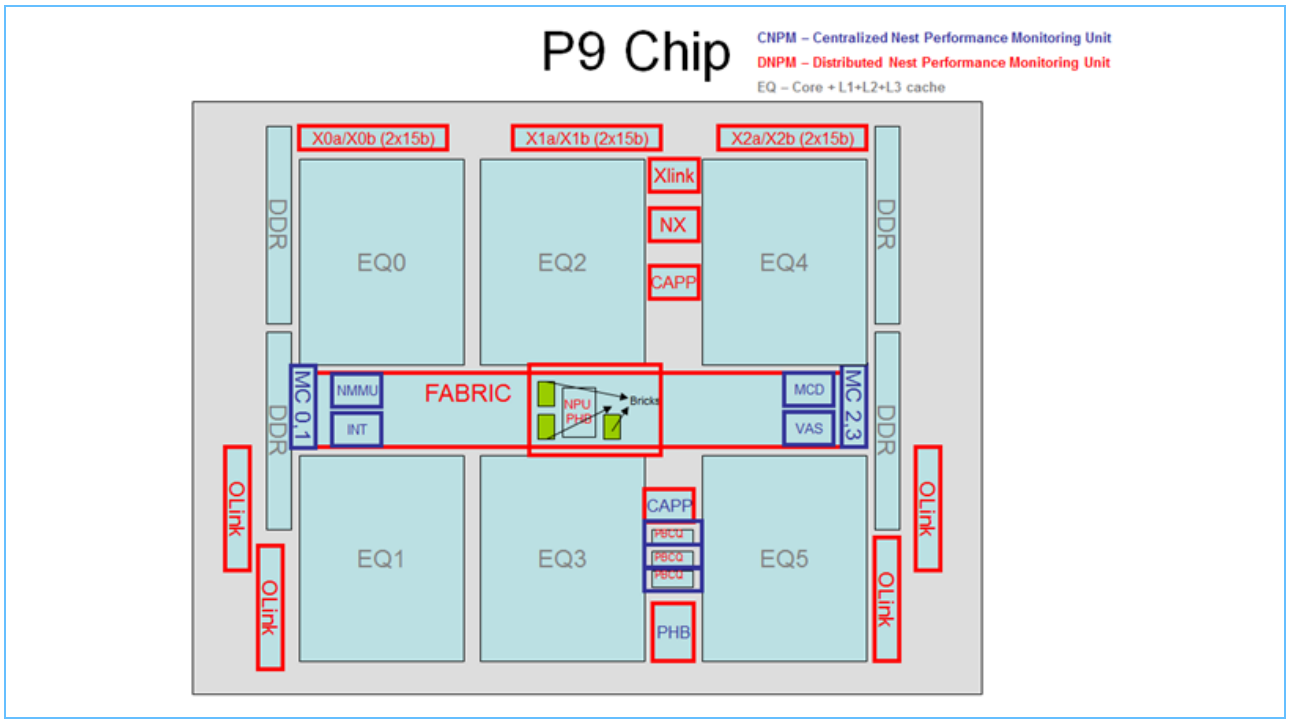
6. Nest PMU Instrumentation and Performance Metrics

6.1 Nest Performance Monitoring Unit Instrumentation

6.1.1 POWER9 Chip Overview

Figure 6-1 shows an overview of the POWER9 chip and the corresponding Nest units and instrumentation.

Figure 6-1. POWER9 Nest PMU Instrumentation



6.1.2 POWER9 Nest PMU Instrumentation

The POWER9 Nest PMU instrumentation spans all chip-level units. These chip-level units are as follows:

- SMP interconnect (PB)
- Memory controller (MC) – SMP interconnect
- Memory buffer asynchronous (MBA) – DIMM interface
- Coherently attached processor proxy (CAPP)
- Xlinks (links connecting across sockets)
- PBCQ (PCIe – SMP interconnect interface)
- PCIe host bridge (PHB)
- NPU/NVLINK (links connecting to NVIDIA GPUs)
- Virtual accelerator switchboard (VAS)
- External interrupt virtualizer engine (XIVE - INT)
- Memory coherency directory (MCD)
- Nest memory management unit (NMMU)
- On-chip accelerator (NX)

6.1.3 Nest Instrumentation Counters (PMUlets)

Nest PMU events are counted on dedicated PMU counters, which are different than thread-level PMU counters (PMC1 - PMC6). The Nest PMU counter (PMUlet) is a 64-bit register, which is partitioned internally as four, 16-bit counters. Each 16-bit counter is an independent counter with individual control. Each PMUlet can be configured to freeze or free run on an overflow.

Unlike the thread level PMC1 - PMC6 counters, the PMUlets are not user accessible. The PMUlets are pre-programmed to count a specific set of events.

Based on the mode in which the Nest IMC is configured, events counted by the Nest IMC differ. See *Section 6.2 Nest Unit and Performance Metrics Support* on page 153 for more information.

6.1.4 Nest PMU Instrumentation Categories:

The Nest PMU instrumentation units are grouped into two categories, which are based on the clock domain that the events are in and the type of counters used:

- Centralized nest performance monitor (CNPM) – units whose PMU events are synchronous to the SMP interconnect (processor bus) clock and share the centralized counters. Centralized counters have access to a 32-bit event bus that all units share along with counters (PMUlets) to access them.
- Distributed nest performance monitor (DNPM) – units whose PMU events are asynchronous to the SMP interconnect (processor bus) clock and have independent counters. Distributed counters have dedicated counters (PMUlets) with dedicated event buses.

Table 6-1 shows the Nest units in the CNPM and DNPM groupings. The clock domain column shows each of the Nest unit's PMU instrumentation design clock domain.

Table 6-1. Nest Unit Instrumentation and Grouping (Sheet 1 of 2)

Instrumentation Type	Unit	Clock Domain	Number of Counters	Counter Width
Centralized Nest Performance Monitor	SMP interconnect (PB)	PB clock at 1.6 - 2.4 GHz	32	16-bit prescalar sizes (20, 16, 8, 4)
	Memory controller synchronous (MCS)	PB clock		
	PCIe synchronous (PBCQ)	PB clock		
	Virtual accelerator switchboard (VAS)	PB clock		
	External interrupt virtualizer engine (XIVE - INT)	PB clock		
	MCD synchronous	PB clock		

Table 6-1. Nest Unit Instrumentation and Grouping (Sheet 2 of 2)

Instrumentation Type	Unit	Clock Domain	Number of Counters	Counter Width
Distributed Nest Performance Monitor	CAPP	PB clock	8	16-bit prescalar sizes (20, 16, 8, 4)
	NMMU	PB clock	8	16-bit prescalar sizes (20, 16, 8, 4)
	X Bus (Xlinks)	PB clock	32 for 3 pairs of Xlinks	16-bit prescalar sizes (20, 16, 8, 4)
	NX Accelerator	PB clock	8	16-bit prescalar sizes (20, 16, 8, 4}
	PCIe PHB4	2 GHz	4 counters per PHB port	48 bits (no prescalar)
	MBA (DIMM interface)	DIMM frequency	4 counters per DDR port	16-bit prescalar sizes (20, 16, 8, 4)

6.2 Nest Unit and Performance Metrics Support

The following sections provide a brief introduction to each of the Nest units and the events, performance metrics, and formulas for deriving performance metrics from those events.

There are two modes in which Nest PMU events can be accessible by the user.

- Default In-Memory Collections (IMC): This is the default mode for PMU collection. A set of pre-defined important Nest PMU hardware events are configured to be counted and posted to memory for access (in-memory collection). The interface to access these events is through the standard performance interface.
- Flexible IMC: This is a flexible-mode, chosen by the user to gather a deeper understanding on the performance aspect of a particular nest unit or to debug a specific-performance issue observed during the default IMC mode. In this mode, a different set of pre-defined Nest PMU hardware events is configured by the IMC algorithm, which provides deeper insights into the performance. The newer configured events are counted and posted to memory for access. In this mode, the default IMC events collection is disabled and the flexible IMC is enabled.

  Note: The interface to access flexible IMC mode events is still under development.

Nest IMC events are grouped as follows:

- Group0 – Those events that are counted exclusively in the default IMC mode
- Group1 – Those events that are counted exclusively in the flexible IMC mode
- Group2 – Those events that are counted in both the default IMC mode and the flexible IMC mode

6.2.1 Internal Fabric (SMP Interconnect)

The POWER9 Fabric controller is the underlying hardware used to create a scalable, cache-coherent, multi-processor system. The POWER9 Fabric controller (FBC) provides coherent and non-coherent memory access, I/O operations, interrupt communication, and system-controller communication. The FBC provides all the interfaces, buffering, and sequencing for command and data operations within the storage subsystem. The FBC is integrated on the POWER9 chip, with 12 or 24 processor cores and on the chip memory subsystem. The POWER9 chip has up to six SMP links that can be used to connect to other POWER9 chips.

6.2.2 Internal Fabric PMU Event and Performance Metrics

Table 6-2 lists the events and performance metrics derived from the internal Fabric events. Other events that are part of the Fabric profile exist but are not listed, because they cannot be used to derive any performance metrics by themselves.

Table 6-2. Event and Performance Metrics for Fabric Events

Event Name	Event Description	Performance Metric Name and Formula	Event Group
PM_PB_EVENT_VG_PUMP	Vector group scope operation (locally mastered) on port *n.*	total_vector_group_pumps (per sec) = ((PM_PB_EVENT_VG_PUMP) / PM_PB_CYC) * PB_Freq	Group2
PM_PB_EVENT_LNS_PUMP	Local nodal scope operation (locally mastered) on port *n.*	total_local_node_pumps (per sec) = ((PM_PB_EVENT_LNS_PUMP) / PM_PB_CYC) * PB_Freq	Group2
PM_PB_EVENT_GROUP_ PUMP	Group scope operation (locally mastered) on port *n.*	total_vector_group_pumps_retries (per sec) = ((PM_PB_EVENT_GROUP_PUMP) / PM_PB_CYC) * PB_Freq	Group2
PM_PB_EVENT_RNS_PUMP	Remote nodal scope operation (locally mastered) on port *n.*	total_local_node_pumps_retries (per sec) = ((PM_PB_EVENT_RNS_PUMP) / PM_PB_CYC) * PB_Freq	Group1
PM_PB_EVENT_RTY_VG_ PUMP	Retry of a vector group scope operation (locally mastered). Retry due to (rty_dropped_rcmd, rty_lpc, rty_other).	total_vector_group_pumps_retries (per sec) = ((PM_PB_EVENT_RTY_VG_PUMP) / PM_PB_CYC) * PB_Freq	Group2
PM_PB_EVENT_RTY_LNS_ PUMP	Retry of a nodal scope operation (locally mastered). Retry due to (rty_dropped_rcmd, rty_lpc, rty_other).	total_local_node_pumps_retries (per sec) = ((PM_PB_EVENT_RTY_LNS_PUMP) / PM_PB_CYC) * PB_Freq	Group2
PM_PB_EVENT_RTY_ GROUP_PUMP	Retry of a group scope operation (locally mastered). Retry due to (rty_dropped_rcmd, rty_lpc, rty_other).	total_group_pumps_retries (per sec) = ((PM_PB_EVENT_RTY_GROUP_PUMP) / PM_PB_CYC) * PB_Freq	Group2
PM_PB_EVENT_RTY_RNS_ PUMP	Retry of a remote nodal scope operation (locally mastered). Retry due to (rty_dropped_rcmd, rty_lpc, rty_other).	total_remote_node_pumps_retries (per sec) = ((PM_PB_EVENT_RTY_RNS_PUMP) / PM_PB_CYC) * PB_Freq	Group1
PM_PB_EVENT_INTERNAL_ DATA_XFER	16 × 32-byte octword (OW) data transfer on an internal Fabric horizontal bus.	total_int_pb_bw (gb/s) = ((PM_PB_INT_DATA_XFER * 512) /PM_PB_CYC) * PB_Freq	Group2
PM_PB_EVENT_DATA_XFER	16 × 32-byte OW data transfer.	total_pb_bw (gb/s) = ((PM_PB_EVENT_DATA_XFER * 512) / PM_PB_CYC) * PB_Freq	Group2
PM_PB_EXT_DATA_XFER	16 × 32-byte OW data transfer sent/received on an internal Fabric X or A bus.	total_ext_pb_bw_sent (gb/s) = ((PM_PB_EXT_DATA_XFER * 512) / PM_PB_CYC) * PB_Freq	Group1

6.2.3 Memory Controller (MCS - Internal Fabric Interface)

The POWER9 chip can be directly attached to memory without the requirement of a buffer chip. Each POWER9 processor chip has two logical memory controller units (MC01 and MC23). Each of these can be connected to four DDR4 DRAM ports. Both 64- and 128-byte data transfers are supported (cache-line interleaving on a 32-byte basis).

Table 6-3 lists the memory controller PMU events and the corresponding performance metrics for Group0 setup.

Table 6-3. Memory Controller PMU Events and Performance Metrics (Group0)

Event	Description	Formula	Metric
PM_MCS0_AMO_OP_DISP	The AMO command dispatches in the MCS.	((PM_MCS0_AMO_DISP) / PM_PB_CYC) * PB_Freq	mcs0_amo_req_disp_rate (per sec)
PM_MCS_64B_RD_OR_WR_DISP	Total 64-byte reads and writes dispatched. Read or write can be chosen by the configuration.	((PM_MCS_64B_RD_OR_WR_DISP*64) / PM_PB_CYC) * PB_Freq	total_mcs0_read_or_write_bw (gb/s)
PM_MCS0_64B_RD_DISP	Total 64-byte read data blocks. Pulses once for every two data grant pulses.	((PM_MCS0_64B_RD_DISP* 64) / PM_PB_CYC) * PB_Freq	total_mcs0_read_bw (gb/s)
PM_MCS0_64B_WR_DISP	Total 64-byte write data blocks. Pulses when 64-bytes of write data is available for any command list, based on complete write data.	((PM_MCS0_64B_WR_DISP* 64) / PM_PB_CYC) * PB_Freq	total_mcs0_write_bw (gb/s)

Table 6-4 lists the memory controller PMU events and performance metrics for Group1 setup.

Table 6-4. Memory Controller PMU Events and Performance Metrics (Group1) (Sheet 1 of 2)

Event	Description	Formula	Metric (per second)
PM_MCS_RRTO_DISP	Number of commands related to a read request to own dispatched (speculative or non-speculative).	(PM_MCS_RRTO_DISP/ PM_PB_CYC)* PB_Freq	mcs_rrto_disp_rate
PM_MCS_HPCWR_DISP	HPC_WR dispatched.	(PM_MCS_HPCWR_DISP/ PM_PB_CYC)* PB_Freq	mcs_rrto_disp_rate
PM_MCS_PARTIAL_WR_DISP	Partial write dispatched (dma_pr_w, ci_pr_ooo_w, ci_pr_w).	(PM_MCS_PARTIAL_WR_DISP /PM_PB_CYC)* PB_Freq	mcs_partial_wr_disp_rate

Table 6-4. Memory Controller PMU Events and Performance Metrics (Group1) (Sheet 2 of 2)

Event	Description	Formula	Metric (per second)
PM_MCS_PF_DISP	Number of prefetch operations dispatched.	(PM_MCS_PF_DISP/PM_PB_CYC)* PB_Freq	mcs_pf_cmd_disp_rate
PM_MCS_64B_SPEC_RD	Total number of 64-byte speculative read commands dispatched.	(PM_MCS_64B_SPEC_RD/PM_PB_CYC)* PB_Freq	mcs_64B_spec_disp_rate
PM_MCS_128B_SPEC_RD	Total number of 128-byte speculative read commands dispatched.	(PM_MCS_128B_SPEC_RD/PM_PB_CYC)* PB_Freq	mcs_128B_spec_disp_rate

6.2.3.1 Memory Buffer Asynchronous Unit

The POWER9 memory buffer asynchronous (MBA) PMU instrumentation measures the DRAM activity per port level. These events are also used to formulate performance metrics for bandwidth.

Table 6-5 lists the MBA PMU events and corresponding performance metrics related to Group0 setup.

Table 6-5. MBA PMU Events and Performance Metrics (Group0)

Event	Description	Formula	Metric
PM_MBA0_READ_BYTES	Read count. increments every read and wraps.	(PM_MBA0_READ_BYTES*64/PM_MBA0_CKE_COUNT) * MBA_Freq	mba read BW (GB/s)
PM_MBA0_WRITE_BYTES	Write count. Increments every write and wraps.	(PM_MBA0_WRITE_BYTES*64/PM_MBA0_CKE_COUNT) * MBA_Freq	mba write BW (GB/s)
PM_MBA0_CKE_COUNT	Counts the number of rising edges for a CKE and wraps.		

Table 6-6 lists the MBA PMU events and performance metrics related to Group1 setup.

Table 6-6. MBA PMU Events and Performance Metrics (Group1)

Event	Description	Formula	Metric
PM_MBA_CYC	MBA clock cycles events corresponding to DRAM frequency.		
PM_MBA_ACT_ALL	Number of activations for either reads or writes.	(PM_MBA_ACT_ALL/PM_MBA_CYC) * MBA_Freq	MBA total activation rate (per second)
PM_MBA_CAS_READ	Number of column address strobe (CAS) for read.	(PM_MBA_CAS_READ/PM_MBA_CYC) * MBA_Freq	MBA read CAS rate (per second)
PM_MBA_CAS_WRITE	Number of CAS for write.	(PM_MBA_CAS_WRITE/PM_MBA_CYC) * MBA_Freq	MBA write CAS rate (per second)

6.2.4 Coherently Attached Processor Proxy

The coherently attached processor proxy (CAPP) unit acts as a proxy to the data/line present in the accelerator on an FPGA connected through the PSL. The CAPP-based acceleration unit has an interface to the chip's internal Fabric on one side and an interface to the interconnect to the PSL on the other side. This is the unit that communicates to the PSL and handles the SMP interconnect operations. The coherently attached processor interface (CAPI) is a PCIe interface for data transmission. The framing is native PCIe.

6.2.5 X Links – SMP Links

The off-chip POWER9 Fabric supports up to three pairs of logical SMP links (X0:X2). X0 is a X0_high and X0_low pair. In the 1-hop topology, the intra-group X Bus links connect up to eight processor chips. The Xlinks carry coherency traffic as well as data and are interchangeable as intra-group processor links. The Xlinks can also be configured as aggregate data-only links. The PMU instrumentation design is in the Nest domain. Therefore, all Xlink performance metric computations use the Nest frequency.

Table 6-7 lists the Xlink PMU events and performance metrics for Group0.

Table 6-7. XLink PMU Events and Performance Metrics (Group0)

Event	Description	Formula	Metric
Xlink cycles	Cycles default for Counter0		
PM_XLINK_ANY_RCMD	Number of any reflected command (rcmd) events received or sent through the Xlink.	(PM_XLINK_ANY_RCMD /PM_XLINK_CYCLES)*PB_freq	xlinks_rcmd_rate (per second)
PM_XLINK_DATA_COUNT	Number of 16-byte data blocks sent or received.	(PM_XLINK_DATA_EVENT * 16 / PM_XLINK_CYCLES) *PB_freq	xlinks_data_bw
PM_XLINK_TOTAL_UTIL	Total utilization of the link due to all commands, header, data, rcmd, and presp.	(PM_XLINK_TOTAL_UTIL/ PM_XLINK_CYCLES _AVAILABLE)*100	Total utilization (%)

Table 6-8 lists the Xlink PMU events and performance metrics for Group1.

Table 6-8. XLink PMU Events and Performance Metrics (Group1)

Event	Description	Formula	Metric
Xlink cycles	Cycles default for Counter0		
PM_XLINK_CMD_UTIL	Any command utilization over the link	(PM_XLINK_CMD_UTIL/ PM_XLINK_CYCLES _AVAILABLE)*100	xlinks_cmd_utilization (%)
PM_XLINK_DHDR_DATA_UTIL	Data header and data utilization over the link	(PM_XLINK_DHDR_DATA_ UTIL/ PM_XLINK_CYCLES _AVAILABLE)*100	Data and header utilization (%)
PM_XLINK_TOTAL_UTIL	Total utilization of the link due to all commands, header, data, rcmd, and presp	(PM_XLINK_TOTAL_UTIL/PM_ XLINK_CYCLES _AVAILABLE) *100	Total utilization (%)

6.2.6 PCIe Host Bridge 4 (PHB4)

The POWER9 PHB4 is a reusable building block that implements a single PCIe root complex port and connects to an adapter slot, a PCIe switch, or a PCIe cable connection. The block is part of a larger unit called the PEC, which also contains a PB block that connects to the internal Fabric in the processor. The PHB4 implements all of the functions required to support the Power Systems and the PCIe Gen4 protocol.

Table 6-9 lists the PHB PMU events for group0.

Table 6-9. PHB PMU Events (Group0)

Event	Description	Formula	Metric
PM_PHB_CYC_CNT	Count PHB4 clock cycles running at 2 GHz.		
PM_PHB_DMA_RD_FROM_PCIE	DMA read received from the PCIe link	(PM_PHB_DMA_RD_FROM_PCIE /PM_PHB_CYC_CNT) * PHB_Freq	phb_dma_rd_rate (per sec)
PM_PHB_DMA_WR_FROM_PCIE	DMA write received from the PCIe link	(PM_PHB_DMA_WR_FROM_PCIE /PM_PHB_CYC_CNT) * PHB_Freq	phb_dma_wr_rate (per sec)
PM_PHB_LD_RESP_FROM_PCIE	MMIO load response received from the PCIe link	(PM_PHB_LD_RESP_FROM_PCIE/PM_PHB_CYC_CNT) * 100	phb_mmio_load_rate (per sec)

6.3 Nest IMC Events Grouping

Table 6-10 on page 159 lists the predefined group of events to be monitored by the default IMC mode and their corresponding memory offsets.

Note: Events related to the flexible IMC will be published in a future release of this document.

In the Nest IMC, a DTS file in conjunction with the PCP interface resolves the offset based on the event names. This eliminates the requirement that the user know the offset of each of the events relative to the base.

Table 6-10. Nest Default IMC Groupings (Sheet 1 of 6)

Offset	Events	Group
0x0	Update_count_h1	G1
0x8	PM_PB_EVENT_VG_PUMP	
0x10	PM_PB_EVENT_LNS_PUMP	
0x18	PM_PB_EVENT_GROUP_PUMP	
0x20	PM_PB_EVENT_RNS_PUMP	
0x28	PM_PB_EVENT_RTY_VG_PUMP	
0x30	PM_PB_EVENT_RTY_LNS_PUMP	
0x38	PM_PB_EVENT_RTY_GROUP_PUMP	
0x40	PM_PB_EVENT_RTY_RNS_PUMP	
0x48	PM_MCS23_64B_RD_OR_WR_DISP_PORT01	
0x50	PM_MCS23_64B_RD_DISP_PORT01	
0x58	PM_MCS23_64B_WR_DISP_PORT01	
0x60	PM_MCS23_AMO_OP_DISP__PORT01	
0x68	PM_MCS23_64B_RD_OR_WR_DISP_PORT23	
0x70	PM_MCS23_64B_RD_DISP_PORT23	
0x78	PM_MCS23_64B_WR_DISP_PORT23	
0x80	PM_PB_CYC	
0x88	PM_PB_VG_PUMP_P01	
0x90	PM_PB_LNS_PUMP_P01	
0x98	PM_PB_GROUP_PUMP_P01	
0xA0	PM_PB_RNS_PUMP_P01	
0xA8	PM_PB_INT_DATA_XFER	
0xB0	PM_PB_EXT_DATA_XFER	
0xB8	PM_PB_NNS_PUMP	
0xC0	PM_PB_RTY_NNS_PUMP_P01	
0xC8	PM_MCS01_64B_RD_OR_WR_DISP_PORT01	
0xD0	PM_MCS01_64B_RD_DISP_PORT01	
0xD8	PM_MCS01_64B_WR_DISP_PORT01	
0xE0	PM_MCS01_AMO_OP_DISP_MC23_PORT01	
0xE8	PM_MCS01_64B_RD_OR_WR_DISP_PORT23	
0xF0	PM_MCS01_64B_RD_DISP_PORT23	
0xF8	PM_MCS01_64B_WR_DISP_PORT23	
0x100	PM_PB_CYC2	

Table 6-10. Nest Default IMC Groupings (Sheet 2 of 6)

Offset	Events	Group
0x118	PM_XLINK1_OUT_EVEN_CYC	G2
0x120	PM_XLINK1_OUT_EVEN_ANY_RCMD	
0x128	PM_XLINK1_OUT_EVEN_DATA_COUNT	
0x130	PM_XLINK1_OUT_EVEN_TOTAL_UTIL	
0x138	PM_XLINK1_OUT_ODD_CYC	
0x140	PM_XLINK1_OUT_ODD_ANY_RCMD	
0x148	PM_XLINK1_OUT_ODD_DATA_COUNT	
0x150	PM_XLINK1_OUT_ODD_TOTAL_UTIL	
0x158	PM_XLINK0_IN_EVEN_CYC	
0x160	PM_XLINK0_IN_EVEN_ANY_RCMD	
0x168	PM_XLINK0_IN_EVEN_DATA_COUNT	
0x170	PM_XLINK0_IN_EVEN_TOTAL_UTIL	
0x178	PM_XLINK0_IN_ODD_CYC	
0x180	PM_XLINK0_IN_ODD_ANY_RCMD	
0x188	PM_XLINK0_IN_ODD_DATA_COUNT	
0x190	PM_XLINK0_IN_ODD_TOTAL_UTIL	
0x198	PM_XLINK2_OUT_EVEN_CYC	
0x1A0	PM_XLINK2_OUT_EVEN_ANY_RCMD	
0x1A8	PM_XLINK2_OUT_EVEN_DATA_COUNT	
0x1B0	PM_XLINK2_OUT_EVEN_TOTAL_UTIL	
0x1B8	PM_XLINK2_OUT_ODD_CYC	
0x1C0	PM_XLINK2_OUT_ODD_ANY_RCMD	
0x1C8	PM_XLINK2_OUT_ODD_DATA_COUNT	
0x1D0	PM_XLINK2_OUT_ODD_TOTAL_UTIL	
0x228	PM_NTL0_CLK_CYC	G3
0x230	PM_NTL0_TX_DATA_FLIT	
0x238	PM_NTL0_RX_ANY_FLIT	
0x240	PM_NTL0_RX_DATA_FLIT	
0x248	PM_NTL1_CLK_CYC	
0x250	PM_NTL1_TX_DATA_FLIT	
0x258	PM_NTL1_RX_ANY_FLIT	
0x260	PM_NTL1_RX_DATA_FLIT	
0x268	PM_NTL2_CLK_CYC	
0x270	PM_NTL2_TX_DATA_FLIT	
0x278	PM_NTL2_RX_ANY_FLIT	
0x280	PM_NTL2_RX_DATA_FLIT	
0x288	PM_NTL3_CLK_CYC	

Table 6-10. Nest Default IMC Groupings (Sheet 3 of 6)

Offset	Events	Group
0x290	PM_NTL3_TX_DATA_FLIT	G3
0x298	PM_NTL3_RX_ANY_FLIT	
0x2A0	PM_NTL3_RX_DATA_FLIT	
0x2A8	PM_NTL4_CLK_CYC	
0x2B0	PM_NTL4_TX_DATA_FLIT	
0x2B8	PM_NTL4_RX_ANY_FLIT	
0x2C0	PM_NTL4_RX_DATA_FLIT	
0x2C8	PM_NTL5_CLK_CYC	
0x2D0	PM_NTL5_TX_DATA_FLIT	
0x2D8	PM_NTL5_RX_ANY_FLIT	
0x2E0	PM_NTL5_RX_DATA_FLIT	
0x2E8	PM_ATS_TCE_TRANS_REQ	
0x2F0	PM_ATS_TCE_MISS	
0x2F8	PM_ATS_NO_TRANS_TCE	
0x300	PM_ATS_CACHE_RERUN	
0x308	PM_XTS_ATR_DEMAND_CHECKOUT_MISS	
0x310	PM_XTS_ATR_DEMAND_CHECKOUT	
0x318	PM_XTS_ATSD_TLBI_RCV	
0x320	PM_XTS_ATSD_SENT	
0x338	PM_PHB0_CYC	G4
0x340	PM_PHB0_DMA_RD_FROM_PCIE	
0x348	PM_PHB0_DMA_WR_FROM_PCIE	
0x350	PM_PHB0_LD_RESP_FROM_PCIE	
0x358	PM_PHB1_CYC	
0x360	PM_PHB1_DMA_RD_FROM_PCIE	
0x368	PM_PHB1_DMA_WR_FROM_PCIE	
0x370	PM_PHB1_LD_RESP_FROM_PCIE	
0x378	PM_PHB2_CYC	
0x380	PM_PHB2_DMA_RD_FROM_PCIE	
0x388	PM_PHB2_DMA_WR_FROM_PCIE	
0x390	PM_PHB2_LD_RESP_FROM_PCIE	
0x398	PM_PHB3_CYC	
0x3A0	PM_PHB3_DMA_RD_FROM_PCIE	
0x3A8	PM_PHB3_DMA_WR_FROM_PCIE	
0x3B0	PM_PHB3_LD_RESP_FROM_PCIE	
0x3B8	PM_PHB4_CYC	
0x3C0	PM_PHB4_DMA_RD_FROM_PCIE	

Table 6-10. Nest Default IMC Groupings (Sheet 4 of 6)

Offset	Events	Group
0x3C8	PM_PHB4_DMA_WR_FROM_PCIE	G4
0x3D0	PM_PHB4_LD_RESP_FROM_PCIE	
0x3D8	PM_PHB5_CYC	
0x3E0	PM_PHB5_DMA_RD_FROM_PCIE	
0x3E8	PM_PHB5_DMA_WR_FROM_PCIE	
0x3F0	PM_PHB5_LD_RESP_FROM_PCIE	
0x3F8	Free	
0x400	Free	
0x408	Free	
0x410	Free	
0x418	Free	
0x420	Free	
0x428	Free	
0x430	Free	
0x448	PM_MBA0_READ_BYTES	G5
0x450	PM_MBA0_WRITE_BYTES	
0x458	PM_MBA0_DRAM_CLK_CYC	
0x460	PM_MBA1_READ_BYTES	
0x468	PM_MBA1_WRITE_BYTES	
0x470	PM_MBA1_DRAM_CLK_CYC	
0x478	PM_MBA2_READ_BYTES	
0x480	PM_MBA2_WRITE_BYTES	
0x488	PM_MBA2_DRAM_CLK_CYC	
0x490	PM_MBA3_READ_BYTES	
0x498	PM_MBA3_WRITE_BYTES	
0x4A0	PM_MBA3_DRAM_CLK_CYC	
0x4A8	Free	
0x4B0	Free	
0x4B8	Free	
0x4C0	Free	
0x4C8	Free	
0x4D0	Free	
0x4D8	Free	
0x4E0	Free	
0x4E8	Free	
0x4F0	Free	
0x4F8	Free	

Table 6-10. Nest Default IMC Groupings (Sheet 5 of 6)

Offset	Events	Group
0x500	Free	G5
0x508	Free	
0x510	Free	
0x518	Free	
0x520	Free	
0x528	Free	
0x530	Free	
0x538	Free	
0x540	Free	
0x558	PM_MBA4_READ_BYTES	G6
0x560	PM_MBA4_WRITE_BYTES	
0x568	PM_MBA4_DRAM_CLK_CYC	
0x570	PM_MBA5_READ_BYTES	
0x578	PM_MBA5_WRITE_BYTES	
0x580	PM_MBA5_DRAM_CLK_CYC	
0x588	PM_MBA6_READ_BYTES	
0x590	PM_MBA6_WRITE_BYTES	
0x598	PM_MBA6_DRAM_CLK_CYC	
0x5A0	PM_MBA7_READ_BYTES	
0x5A8	PM_MBA7_WRITE_BYTES	
0x5B0	PM_MBA7_DRAM_CLK_CYC	
0x5B8	Free	
0x5C0	Free	
0x5C8	Free	
0x5D0	Free	
0x5D8	Free	
0x5E0	Free	
0x5E8	Free	
0x5F0	Free	
0x5F8	Free	
0x600	Free	
0x608	Free	
0x610	Free	
0x618	Free	
0x620	Free	
0x628	Free	
0x630	Free	

Table 6-10. Nest Default IMC Groupings (Sheet 6 of 6)

Offset	Events	Group
0x638	Free	G6
0x640	Free	
0x648	Free	
0x650	Free	
0x668	PM_NPCQ0_CREQ_BRICK0	G7
0x670	PM_NPCQ0_DOWNGRADE_REQ_BRICK0	
0x678	PM_NPCQ0_CREQ_BRICK1	
0x680	PM_NPCQ0_DOWNGRADE_REQ_BRICK1	
0x688	PM_NPCQ1_CREQ_BRICK0	
0x690	PM_NPCQ1_DOWNGRADE_REQ_BRICK0	
0x698	PM_NPCQ1_CREQ_BRICK1	
0x6A0	PM_NPCQ1_DOWNGRADE_REQ_BRICK1	
0x6A8	PM_NPCQ2_CREQ_BRICK0	
0x6B0	PM_NPCQ2_DOWNGRADE_REQ_BRICK0	
0x6B8	PM_NPCQ2_CREQ_BRICK1	
0x6C0	PM_NPCQ2_DOWNGRADE_REQ_BRICK1	

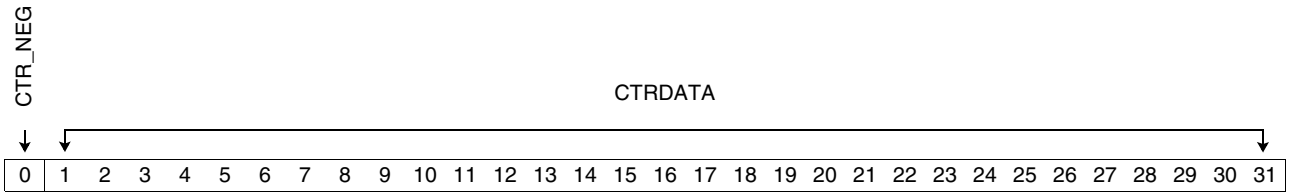
Appendix A. Performance Monitor Registers

This section describes the following performance monitor related registers:

- Performance Monitor Counters (PMC1 - 6)
- Core Monitor Mode Control Register (MMCRC)
- Performance Monitor Control Register 0 (MMCR0)
- Performance Monitor Mode Control Register 1 (MMCR1)
- Performance Monitor Mode Control Register 2 (MMCR2)
- Monitor Mode Control Register A (MMCRA)
- Sampled Instruction Event Register (SIER)
- Sampled Instruction Address Register (SIAR)
- Sampled Data Address Register (SDAR)

A.1 Performance Monitor Counters (PMC1 - 6)

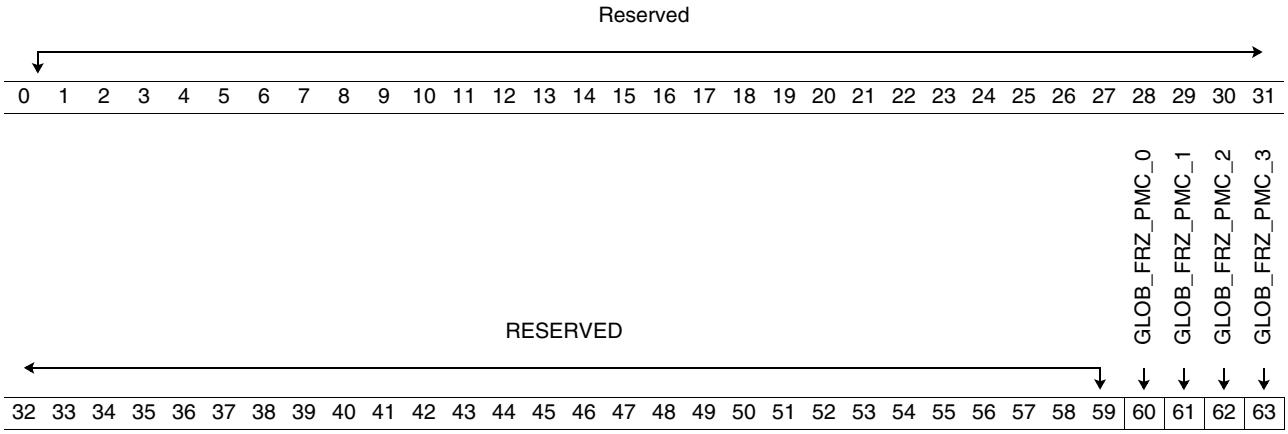
The six performance monitor counters, PMC1 through PMC6, are 32-bit registers that count events. PMC1 - PMC4 are referred to as programmable counters because the events that can be counted can be specified by software. The codes that identify the counted events are selected by specifying the appropriate code in the PMCxSEL event select fields in MMCR1[32:63]. Chaining of two or more counters is accomplished by setting a PMCxSEL to select an event for counting defined as the overflow (Msb going to '1') of a chained counter. The counted events are described *Section 5 Core PMU Events* on page 34. PMC5 and PMC6 are not programmable. PMC5 counts instructions completed, and PMC6 counts cycles.



Bits	Field Name	Description
0	CTR_NEG, CTRDATA	Counter negative bit. When an adjacent PMC uses overflow counting, this becomes counter data.
1:31	CTRDATA	Counter data.

A.2 Core Monitor Mode Control Register (MMCRC)

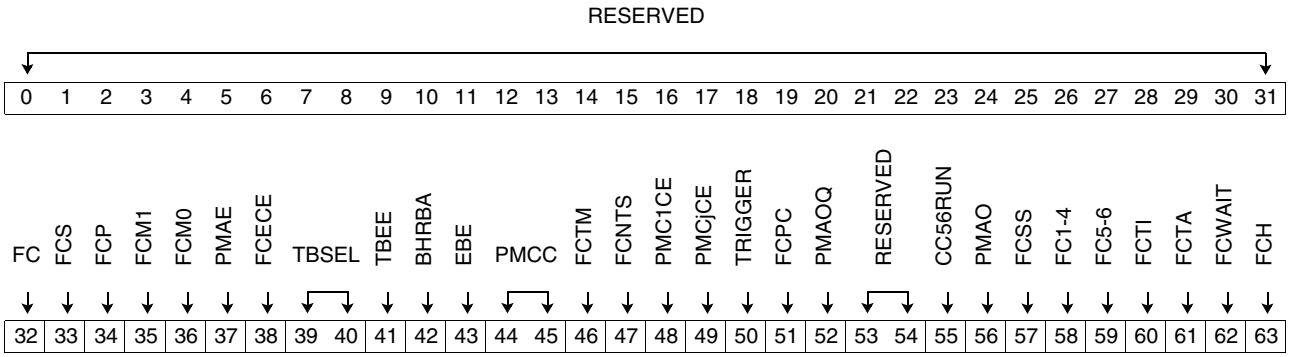
This SPR is not replicated on a per-split core basis. This is a core-level SPR (hypervisor access only) that is used for a variety of purposes.



Bits	Field Name	Description
0:59	RESERVED	Reserved.
60	GLOB_FRZ_PMC_0	Global freeze for thread level PMU Thread0. **Note:** When MMCR0[PMCC] = '11', this bit affects PMCs 1 - 4; otherwise, it controls PMCs 1 - 6. In 4LPAR mode, this maps to LPAR0.
61	GLOB_FRZ_PMC_1	Global freeze for thread level PMU Thread1. **Note:** When MMCR0[PMCC] = '11', this bit affects PMCs 1 - 4; otherwise, it controls PMCs 1 - 6. In 4LPAR mode, this maps to LPAR1.
62	GLOB_FRZ_PMC_2	Global freeze for thread level PMU Thread2. **Note:** When MMCR0[PMCC] = '11', this bit affects PMCs 1 - 4; otherwise, it controls PMCs 1 - 6. In 4LPAR mode, this maps to LPAR2.
63	GLOB_FRZ_PMC_3	Global freeze for thread level PMU Thread3. NOTE: When MMCR0[PMCC] = 11, this bit affects PMCs 1 - 4 otherwise, it controls PMCs 1 - 6. In 4LPAR mode, this maps to LPAR3.

A.3 Performance Monitor Control Register 0 (MMCR0)

The 64-bit Performance Monitor Mode Control Register 0 (MMCR0) controls the basic operation (start/stop/freeze) of the performance monitor.



Bits	Field Name	Description
0:31	Reserved	Reserved.
32	FC	Freeze counters. 0 The PMCs are incremented (if permitted by other MMCR bits). 1 The PMCs are not incremented. The processor sets this bit to '1' when an enabled condition or event occurs and the "freeze counter on an enabled condition or an event bit" is '1' (MMCR0[FCECE] = '1'). **Note:** When MMCR0[PMCC] = '11', this bit affects PMCs 1 - 4; otherwise, it controls PMCs 1 - 6.
33	FCS	Freeze counters and the BHRB are in privileged state. 0 The PMCs are incremented, and entries are written into the BHRB (if permitted by other MMCR bits). 1 The PMCs are not incremented, and entries are not written into the BHRB if MSR[HV, PR] = '00'. **Note:** When MMCR0[PMCC] = '11', this bit affects PMCs 1 - 4; otherwise, it controls PMCs 1 - 6.
34	FCP	Freeze counters and the BHRB are in problem state. If MMCR0[FCPC] = '0' and FCP is equal to: 0 The PMCs are incremented, and entries are written into the BHRB (if permitted by other MMCR bits). 1 The PMCs are not incremented, and entries are not written into the BHRB if MSR[PR] = '1'. If MMCR0[FCPC] = '1' and FCP is equal to: 0 The PMCs are not incremented, and entries are not written into the BHRB if MSR[HV, PR] = '01'. 1 The PMCs are not incremented, and entries are not written into the BHRB if MSR[HV, PR] = '11'. **Notes:** 1. To freeze the counters in problem state regardless of MSR[HV], MMCR0[FCPC] must be set to '0' and MMCR0[FCP] must be set to '1'. 2. When MMCR0[PMCC] = '11', this bit affects PMCs 1 - 4; otherwise, it controls PMCs 1 - 6.
35	FCM1	Freeze counters when the performance monitor Mark bit (MSR[PMM]) is set to '1'. 0 The PMCs are incremented. 1 The PMCs are not incremented when the MSR mark bit is '1' (MSR[PMM] = '1'). **Note:** When MMCR0[PMCC] = '11', this bit affects PMCs 1 - 4; otherwise, it controls PMCs 1 - 6.

Bits	Field Name	Description
36	FCM0	Freeze counters when the performance monitor Mark bit (MSR[PMM]) is set to '0'. 0 The PMCs are incremented. 1 The PMCs are not incremented when MSR[PMM] = '0'. **Note:** When MMCR0[PMCC] = '11', this bit affects PMCs 1 - 4; otherwise, it controls PMCs 1 - 6.
37	PMAE	Performance monitor alert enable. 0 Performance monitor alerts are disabled. 1 Performance monitor alerts are enabled until a performance monitor alert occurs, at which time MMCR0[PMAE] is set to '0' and MMCR0[PMA0] is set to '1'. When MMCR0[EBE] = '1', setting BESCR[PME] to '0' or '1' sets PMAE to '0' or '1' respectively. For implementations that do not provide a performance monitor interrupt, software can set PMAE to '1' and then poll the bit to determine whether an enabled condition or event has occurred. **Note:** Software can set this bit and MMCR0[PMAO] to '0' to prevent performance monitor exceptions. Software can set this bit to '1' and then poll the bit to determine whether an enabled condition or event has occurred. This is especially useful for software that runs with MSR[EE] = '0'. In earlier versions of the architecture that lacked the concept of performance monitor alerts, this bit was called the Performance Monitor Exception Enable (PMXE). **Note:** When MMCR0[PMCC] = '11', this bit affects PMCs 1 - 4; otherwise, it controls PMCs 1 - 6.
38	FCECE	Freeze counters on enabled condition or event. 0 The PMCs are incremented (if permitted by other MMCR bits). 1 The PMCs are incremented (if permitted by other MMCR bits) until an enabled condition or event occurs when MMCR0[TRIGGER] = '0', at which time MMCR0[FC] is set to '1'. If the enabled condition or event occurs when MMCR0[TRIGGER] = '1', the FCECE bit is treated as if it were '0'. **Note:** When MMCR0[PMCC] = '11', this bit affects PMCs 1 - 4; otherwise, it controls PMCs 1 - 6.
39: 40	TBSEL	Timebase selector. 00 Timebase bit 47 is selected. 01 Timebase bit 51 is selected. 10 Timebase bit 55 is selected. 11 Timebase bit 63 is selected. When the selected timebase transitions from '0' to '1', the timebase event is enabled (MMCR0[TBEE] = '1'), and the performance monitor interrupt is enabled: a performance monitor interrupt occurs and the performance monitor interrupt is disabled (MMRC0[PMAE] is negative). In multiprocessor systems with the timebase registers synchronized among the processors, timebase transition events can be used to correlate the performance monitor data obtained by the several processors provided that software has specified the same TBSEL value for all of the processors in the system. Even with multi-LPAR support, LPAR0 timebase bits are used so that all partitions are counting events over the same periods and throwing alerts at the same time. **Note:** This bit is the default for timebase selection. However, other bits can be selected in the timebase unit.
41	TBEE	Timebase exception enable. 0 Disable timebase transition events. 1 Enable timebase transition events.
42	BHRBA	BHRB access. This field controls whether the BHRB instructions are available in problem state. If an attempt is made to execute a BHRB instruction in problem state when the BHRB instructions are not available, a Facility Unavailable interrupt occurs. 0 **mfbhrb** and **clrbhrb** are not available in problem state. 1 **mfbhrb** and **clrbhrb** are available in problem state.

Bits	Field Name	Description
43	EBE	Performance monitor event-based branch enable. This field controls whether performance monitor event-based branches are enabled. 0 Performance monitor event-based branches are disabled. 1 Performance monitor event-based branches are enabled. **Note:** Enabling event-based branches gives problem state programs visibility to and control of MMCR0[PMAE] and MMCR0[PMAO].This enables problem state programs to recognize when performance monitor event-based exceptions have occurred and to re-enable performance monitor event-based exceptions. To enable a problem state program to use the event-based branch facility for performance monitor events, software first initializes the performance monitor registers to values appropriate to the program, sets MMCR0[PMAE] and MMCR0[PMAO] to '0', sets BESCR to '0', and sets MMCR0[EBE] to '1'. MMCR0[PMCC] should also be set to '10' or '11' to give problem state programs write access to the PMCs. If the event-based branch facility has not been enabled in the Facility Status and Control Register (FSCR) and Hypervisor Facility Status and Control Register (HFSCR), it must be enabled in these registers as well. **Note:** When MMCR0[PMCC] = '11', this bit affects PMCs 1 - 4; otherwise, it controls PMCs 1 - 6.
44:45	PMCC	PMC control. This field controls the number of counters that are included in the performance monitor and the set of registers that is available to be read and written in problem state. If an attempt is made to read a register that is unavailable to be read, or an attempt is made to write a register that is unavailable to be written to in problem state, a facility unavailable interrupt occurs. The values and their meanings are described as follows: 00 PMCs 1 - 6 are included in the performance monitor, and SPRs 768 - 782 are available to be read in problem state but are not available to be written in problem state. 01 PMCs 1 - 6 are included in the performance monitor, but SPRs 768 - 782 are not available to be read or written in problem state. 10 PMCs 1 - 6 are included in the performance monitor. Selected fields of MMCR0, MMCR2, MMCRA, and all bits of PMCs 1 - 6 are available to be read and written in problem state. SIER, SDAR, and SIAR are read-only in problem state. All other SPRs in the range of 768 - 782 are not available to be read or written in problem state. 11 PMCs 5 - 6 are not included in the performance monitor. Group A SPRs (MMCR2, MMCRA, PMC 1 - 6, MMCR0) except for PMCs 5 - 6 can be read and written in problem state. Group B (SIER, SIAR, SDAR, MMCR1), except for MMCR1, can be read in problem state. If an attempt is made when in problem state, to read or write to PMCs 5 - 6 or to read from MMCR1, a Facility Unavailable interrupt occurs. When an SPR is made available by the PMCC field, it is available only if it has not been made unavailable by the HFSCR. When PMCs 5 and 6 are not part of the performance monitor (for example, when PMCC = '11'), counter negative conditions in PMCs 5 and 6 do not result in performance monitor alerts or exceptions and do not result in performance monitor interrupts
46	FCTM	Freeze counters when transactional memory (TM) is in the transactional state (MSR[TS] = '10'). 0 PMCs are incremented. 1 PMCs are not incremented when TM is in a transactional state. **Note:** When MMCR0[PMCC] = '11', this bit affects PMCs 1 - 4; otherwise, it controls PMCs 1 - 6.
47	FCNTS	Freeze counters when transactional memory is in nontransactional state (MSR[TS] = '00'). 0 PMCs are incremented (if permitted by other MMCR bits). 1 PMCs are not incremented when TM is in a non-transactional state. **Note:** When MMCR0[PMCC] = '11', this bit affects PMCs 1 - 4; otherwise, it controls PMCs 1 - 6.
48	PMC1CE	PMC1 condition enable. This bit determines whether the counter negative condition due to a negative value in PMC1 is enabled. 0 Disable PMC1 counter negative condition. 1 Enable PMC1 counter negative condition.

Bits	Field Name	Description
49	PMCjCE	PMCj condition enable. This bit determines whether the counter negative condition, due to a negative value in PMCj ($2 \le j$), is enabled. 0 Disable PMCj ($2 \le j$) counter negative condition. 1 Enable PMCj ($2 \le j$) counter negative condition. **Note:** When MMCR0[PMCC] = '11', this bit affects PMCs 2 - 4. Otherwise, it controls PMCs 2 - 6. When MMCR0[PMCC] = '11', PMC refers to PMCs 1 - 4 and PMCj or PMCjCE refers to PMCj or PMCjCE, respectively, where j = 2 - 4. Otherwise, PMC refers to PMCs 1 - 6 and PMCj or PMCjCE refers to PMCj or PMCjCE, respectively, where j = 2 - 6.
50	TRIGGER	TRIGGER enable. 0 The PMCs are incremented (if permitted by other MMCR bits). 1 PMC1 is incremented (if permitted by other MMCR bits). The PMCjs are not incremented until PMC1 is negative or an enabled condition or event occurs, at which time the PMCjs resume incrementing (if permitted by other MMCR bits). MMCR0[TRIGGER] is set to '0'. **Note:** See the description of the FCECE bit regarding the interaction between TRIGGER and FCECE. Resume counting in the PMCjs when PMC1 becomes negative, without causing a performance monitor interrupt. Then freeze all PMCs (and optionally cause a performance monitor interrupt) when a PMCj becomes negative. The PMCjs then reflect the events that occurred between the time PMC1 became negative and the time a PMCj becomes negative. This use requires the following MMCR0 bit settings: • TRIGGER = '1' • PMC1CE = '0' • PMCjCE = '1' • TBEE = '0' • FCECE = '1' • PMAE = '1' (if a performance monitor interrupt is required) Resume counting in the PMCjs when PMC1 becomes negative and cause a performance monitor interrupt without freezing any PMCs. The PMCjs then reflect the events that occurred between the time PMC1 became negative and the time the interrupt handler reads them. This use requires the following MMCR0 bit settings: • TRIGGER = '1' • PMC1CE = '1' • TBEE = '0' • FCECE = '0' • PMAE = '1' **Note:** When MMCR0[PMCC] = '11', this bit affects PMCs 1 - 4; otherwise, it controls PMCs 1 - 6.
51	FCPC	Freeze counters and the BHRB in the problem state condition. This bit controls the operation of bit 34 (FCP). See the definition of bit 34 for details. **Note:** When MMCR0[PMCC] = '11', this bit affects PMCs 1 - 4; otherwise, it controls PMCs 1 - 6.
52	PMAOQ	0 PMU Interrupt is asynchronous. 1 PMU Interrupt is synchronous.
53:54	RESERVED	Reserved.
55	CC56RUN	Freeze counters 5 and 6 in the wait state. When MMCR0[PMCC] = '11', the setting of this bit has no effect; otherwise, it is defined as follows: 0 PMCs 5 and 6 are incremented only when CTRL[RUN] = '1' (if permitted by other MMCR bits). Software is expected to set CTRL[RUN] = '0' when it is in a wait state; for example, when there is no process ready to run. 1 PMCs 5 and 6 are incremented regardless of the state of CTRL[RUN]. **Note:** In the IBM POWER8 processor (DD1.0), this bit was FC56WAIT and its behavior was the inverse of CC56RUN.

Bits	Field Name	Description
56	PMAO	Performance monitor alert has occurred. 0 A performance monitor event has not occurred since the last time software set this bit to '0'. 1 A performance monitor event has occurred since the last time software set this bit to '0'. This bit is set to '1' by the processor when a performance monitor event occurs. This bit can be set to '0' only by the **mtspr** instruction. When MMCR0[EBE] = '1', setting BESCR[PME0] to '0' or '1' sets PMA0 to '0' or '1', respectively. Software can set this bit to '1' to simulate the occurrence of a performance monitor event. Software should set this bit to '0' after handling the performance monitor event. **Note:** When MMCR0[PMCC] = '11', this bit affects PMCs 1 - 4; otherwise, it controls PMCs 1 - 6.
57	FCSS	Freeze counters when transactional memory is in the suspended state (MSR[TS] = '01'). 0 PMCs are incremented (if permitted by other MMCR bits). 1 PMCs are not incremented when transactional memory is in the suspended state. **Note:** When MMCR0[PMCC] = '11', this bit affects PMCs 1 - 4; otherwise, it controls PMCs 1 - 6.
58	FC1-4	Freeze counters 1 - 4. 0 PMC 1 - 4 are incremented (if permitted by other MMCR bits). 1 PMC 1 - 4 are not incremented.
59	FC5-6	Freeze counters 5 - 6. 0 PMC5 - 6 are incremented (if permitted by other MMCR bits). 1 PMC5 - 6 are not incremented. **Note:** When MMCR0[PMCC] = '11', this bit has no effect.
60	FCTI	Freeze counters in tags inactive mode. 0 The PMCs are incremented. 1 The PMCs are not incremented in tags inactive mode. **Note:** When MMCR0[PMCC] = '11', this bit affects PMCs 1 - 4; otherwise, it controls PMCs 1 - 6.
61	FCTA	Freeze counters in tags active mode. 0 The PMCs are incremented. 1 The PMCs are not incremented in tags active mode. **Note:** When MMCR0[PMCC] = '11', this bit affects PMCs 1 - 4; otherwise, it controls PMCs 1 - 6.
62	FCWAIT	Freeze counters in the wait state. 0 The PMCs are incremented (if permitted by other MMCR bits). 1 The PMCs are not incremented if CTRL[RUN] = '0'. Software is expected to set CTRL[RUN] = '0' when it is in a wait state; for example, when there is no process ready to run. **Notes:** 1. When MMCR0[PMCC] = '11', this bit affects PMCs 1 - 4; otherwise, it controls PMCs 1 - 6. 2. PM_CYC counts cycles regardless of the run latch.
63	FCH	Freeze counters and BHRB in a hypervisor state. 0 The PMCs are incremented (if permitted by other MMCR bits) and BHRB entries are written (if permitted by MMCRA bits). 1 If MSR[HV, PR] = '10', the PMCs are not incremented and the BHRB entries are not written. **Note:** When MMCR0[PMCC] = '11', this bit affects PMCs 1 - 4; otherwise it controls PMCs 1 - 6.

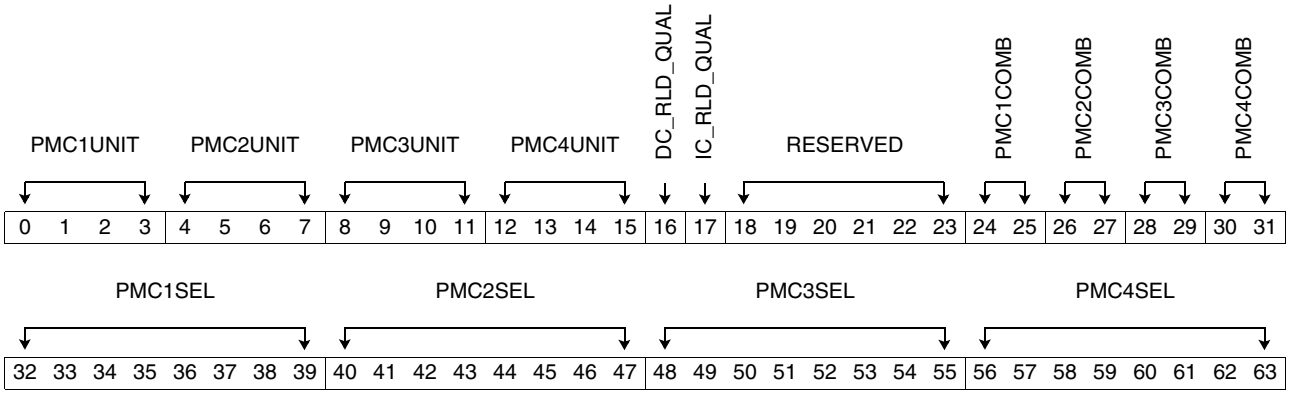
Table A-1. shows the MMCR0 freeze logic based on the MSR state.

Table A-1. MMCR0 Freeze Logic per MSR State, FCH, FCS, FCP, and FCPC

FCH	FCS	FCPC	FCP	MSR [HV]	MSR [PR]	Freeze Condition Description
x	x	0	1	x	1	PMCs freeze when PR = '1'(irrespective of HV). MSR[PR] = '1'.
x	x	1	0	0	1	PMCs freeze when in true problem state. MSR[HV, PR] = '01'.
x	x	1	1	1	1	PMCs freeze in adjunct state. MSR[HV, PR] = '11'.
x	1	x	x	0	0	PMCs freeze in privileged state. MSR[HV, PR] = '00'.
1	x	x	x	1	0	PMCs freeze in hypervisor state. MSR[HV, PR] = '10'.

A.4 Performance Monitor Mode Control Register 1 (MMCR1)

The 64-bit Performance Monitor Mode Control Register 1 (MMCR1) enables software to specify the events that are counted by the PMCs.



Bits	Field Name	Description
0:3	PMC1UNIT	PMC1 events unit selector. 0000 Reserved 0001 Reserved 0010 ISU0 0011 ISU1 0100 IFU0 0101 IFU1 0110 L2/L3 Bus0 0111 Reserved 1000 MMU1 1001 MMU2 1010 MMU3 1011 MMU4 1100 LSU0 1101 LSU1 1110 LSU2 1111 LSU3 Refer to *Table A-3.* on page 175.
4:7	PMC2UNIT	PMC2 events unit selector (same encoding as PMC1UNIT). Refer to *Table A-3.* on page 175.
8:11	PMC3UNIT	PMC3 events unit selector (same encoding as PMC1UNIT). Refer to *Table A-3.* on page 175.
12:15	PMC4UNIT	PMC4events unit selector (same encoding as PMC1UNIT). Refer to *Table A-3.* on page 175.
16	DC_RLD_QUAL	This bit defines the data-cache reload qualifier. 0 Counts only demand reloads to the L1 data cache. 1 Counts all reloads to the L1 data cache (demand and prefetch).
17	IC_RLD_QUAL	This bit defines the instruction-cache reload qualifier. 0 Counts only demand reloads to the L1 instruction cache. 1 Counts all reloads to the L1 instruction cache (demand and prefetch).
18:23	RESERVED	Reserved.

Bits	Field Name	Description
24:25	PMC1COMB	This bit controls the event bus combinatorial function for PMC1: 00 Selects event bus bit 0. 10 Selects event bus bit 1. 01 AND the event bus bits 0 and 1. 11 ADD the event bus bits 0 and 1.
26:27	PMC2COMB	This bit controls the event bus combinatorial function for PMC2 (see PMC1COMB bit description).
28:29	PMC3COMB	This bit controls the event bus combinatorial function for PMC3 (see PMC1COMB).
30:31	PMC4COMB	This bit controls the event bus combinatorial function for PMC4 (see PMC1COMB).
32:39	PMC1SEL	PMC1 event selector bits 1 - 8. The value in this bit field combined with MMCR1[0:31] determines which event will be counted. x'10' PMC1 counts PMC4 overflows. x'24' PMC1 counts PMC5 overflows. When counting overflows, freeze conditions for both counters must be identical. See *Table A-2.* on page 175 for selection of event types. PMCxSEL(1:2) = '10' selects event bus events. PMCxSEL(1:3) = '111' for compatibility direct events. PMCxSEL(1) = '0' for other direct events. When PMCxSEL(8) is high, the falling edges of an event are counted rather than the event itself. Changes from nonzero to '0' are the "edge" for event bus events.
40:47	PMC2SEL	PMC2 event selector bits 1 - 8. The value in this bit field combined with MMCR1[0:31] determines which event will be counted. x'10' PMC2 counts PMC1 overflows. When counting overflows, freeze conditions for both counters must be identical. See *Table A-2.* on page 175 for selection of event types. PMCxSEL(1:2) = '10' selects event bus events. PMCxSEL(1:3) = '111' for compatibility direct events. PMCxSEL(1) = '0' for other direct events. When PMCxSEL(8) is high, the falling edges of an event are counted rather than the event itself. Changes from nonzero to '0' are the "edge" for event bus events.
48:55	PMC3SEL	PMC3 event selector bits 1 - 8. The value in this bit field combined with MMCR1[0:31] determines which event will be counted. x'10' PMC3 counts PMC2 overflows. x'24' PMC3 counts PMC6 overflows. When counting overflows, freeze conditions for both counters must be identical. See *Table A-2.* on page 175 for selection of event types. PMCxSEL(1:2) = '10' selects event bus events. PMCxSEL(1:3) = '111' for compatibility direct events. PMCxSEL(1) = '0' for other direct events. When PMCxSEL(8) is high, falling edges of an event are counted rather than the event itself. Changes from non-zero to 0 are the "edge" for event bus events.
56:63	PMC4SEL	PMC4 event selector bit 1 - 8. The value in this bit field combined with MMCR1[0:31] determines which event will be counted. x'10' PMC4 counts PMC3 overflows. When counting overflows, freeze conditions for both counters must be identical. See *Table A-2.* on page 175 for selection of event types. PMCxSEL(1:2) = '10' selects event bus events. PMCxSEL(1:3) = '111' for compatibility direct events. PMCxSEL(1) = '0' for other direct events, When PMCxSEL(8) is high, the falling edges of an event are counted rather than the event itself. Changes from nonzero to '0' are the "edge" for event bus events.

Notes:

1. Event counting on PMC1 - 4 suspends counting for one cycle during SPR writes to MMCR1 to avoid a spurious or phantom count during the transition.
2. For the POWER9 processor, the PMC1 - 4 selector has been increased from 8 bits to 9 bits. For compatibility events, bit 0 of the selector is not used.

Table A-2. lists the PMC event selector bit values versus the event type.

Table A-2. MMCR1[PMCxSEL] Selection of Direct Events versus Event Bus Events

PMCxSEL(0:3)	Event Type
0000	Direct Events
0001	
0010	
0011	
0100	
0101	
0110	
0111	
1000	Event Bus Events
1001	
1010	
1011	
1100	Reserved
1101	
1110	Compatibility Direct Events
1111	

Table A-3. lists the PMC events unit selector values versus the physical event bus selection.

Table A-3. MMCR1[PMCxUNIT] Selection of Physical Event Buses (Sheet 1 of 2)

PMCxUNIT(0:3)	Physical Event Bus Selected
0000	Reserved
0001	
0010	if_pc_pmon_events
0011	
0100	
0101	
0110	l2_pc_pmon_l2event_bus0
0111	l2_pc_pmon_l2event_bus1
1000	l2_pc_pmon_l3event_bus0
Note: VSU -> ISU -> IFU -> PMU core signals arrive on if_pc_pmon_events bus to pc.	

Table A-3. MMCR1[PMCxUNIT] Selection of Physical Event Buses (Sheet 2 of 2)

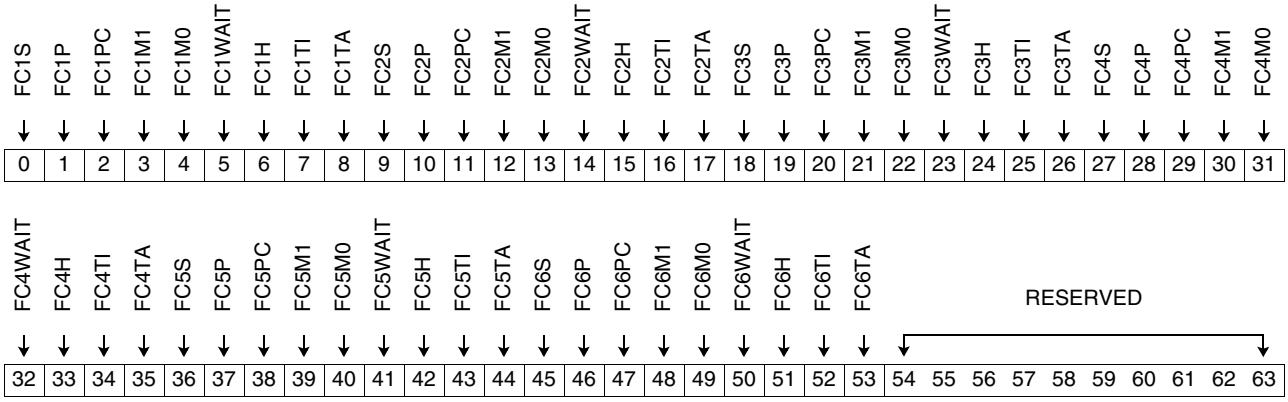
PMCxUNIT(0:3)	Physical Event Bus Selected
1001	l2_pc_pmon_l3event_bus1
1010	if_pc_pmon_events
1011	
1100	
1101	
1110	
1111	

Note: VSU -> ISU -> IFU -> PMU core signals arrive on if_pc_pmon_events bus to pc.

A.5 Performance Monitor Mode Control Register 2 (MMCR2)

The 64-bit Performance Monitor Mode Control Register 2 (MMCR2) contains 9-bit fields that control the operation of PMCs 1 - 6.

A single bit in each Cn field of MMCR2 is described in the following bit description table. When MMCR0[PMCC] = '11', fields C1 - C4 control the operation of PMC1 - PMC4, respectively and fields C5 and C6 are meaningless; otherwise, fields C1 - C6 control the operation of PMC1 - PMC6, respectively. The bit definitions of each Cn field are as follows, where n = 1,...6.



Bits	Field Name	Description
0	FC1S	Freeze counter n in privileged state. 0 PMC1 is incremented (if permitted by other MMCR bits). 1 PMC1 is not incremented if MSR[HV, PR] = '00'.
1	FC1P	Freeze counter n in problem state if MSR[HV] = '0'. 0 PMC1 is incremented (if permitted by other MMCR bits). 1 If FCnPC = 0, then PMC1 is not incremented if MSR[HV, PR] = '1'.
2	FC1PC	Freeze counter n in problem state if MSR[HV] = '1'. 0 PMC1 is incremented (if permitted by other MMCR bits). 1 If FCnP = '0', PMC1 is not incremented if MSR[HV, PR] = '01' (true problem state). If PCnP = '1', PMCn is not incremented if MSR[HV, PR] = '11' (adjunct state).
3	FC1M1	Freeze counter n while Mark = '1'. 0 PMC1 is incremented (if permitted by other MMCR bits). 1 PMC1 is not incremented if MSR[PMM] = '1'.
4	FC1M0	Freeze counter n while Mark = '0'. 0 PMC1 is incremented (if permitted by other MMCR bits). 1 PMC1 is not incremented if MSR[PMM] = '0'.
5	FC1WAIT	Freeze counter n in wait state. 0 PMC1 is incremented (if permitted by other MMCR bits). 1 PMC1 is not incremented if CTRL[RUN] = '0'. Software is expected to set CTRL[RUN] = '0' when it is in a wait state; for example, when there is no process ready to run. **Note:** PM_CYC counts cycles regardless of the run latch.
6	FC1H	Freeze counter n in hypervisor state. 0 PMC1 is incremented (if permitted by other MMCR bits). 1 PMC1 is not incremented if MSR[HV, PR] = '10'.
7	FC1TI	Freeze counter n in tags inactive mode. 0 PMC1 is incremented (if permitted by other MMCR bits). 1 PMC1 is not incremented in tags inactive mode.
8	FC1TA	Freeze counter n in tags active mode. 0 PMC1 is incremented (if permitted by other MMCR bits). 1 PMC1 is not incremented in tags active mode.
9	FC2S	Freeze counter n in privileged state. 0 PMCn is incremented (if permitted by other MMCR bits). 1 PMCn is not incremented if MSR[HV, PR] = '00'.
10	FC2P	Freeze counter n in problem state if MSR[PR] = '1'. 0 PMCn is incremented (if permitted by other MMCR bits). 1 If FCnPC = '0', PMCn is not incremented if MSR[PR] = '1'.
11	FC2PC	Freeze counter n in problem state if MSR[HV] = '1'. 0 PMCn is incremented (if permitted by other MMCR bits). 1 If FCnP = '0', PMCn is not incremented if MSR[HV, PR] = '01' (true problem state). If PCnP = '1', PMCn is not incremented if MSR[HV, PR] = '11' (adjunct state).
12	FC2M1	Freeze counter n while Mark = '1'. 0 PMCn is incremented (if permitted by other MMCR bits). 1 PMCn is not incremented if MSR[PMM] = '1'.
13	FC2M0	Freeze counter n while Mark = '0'. 0 PMCn is incremented (if permitted by other MMCR bits). 1 PMCn is not incremented if MSR[PMM] = '0'.

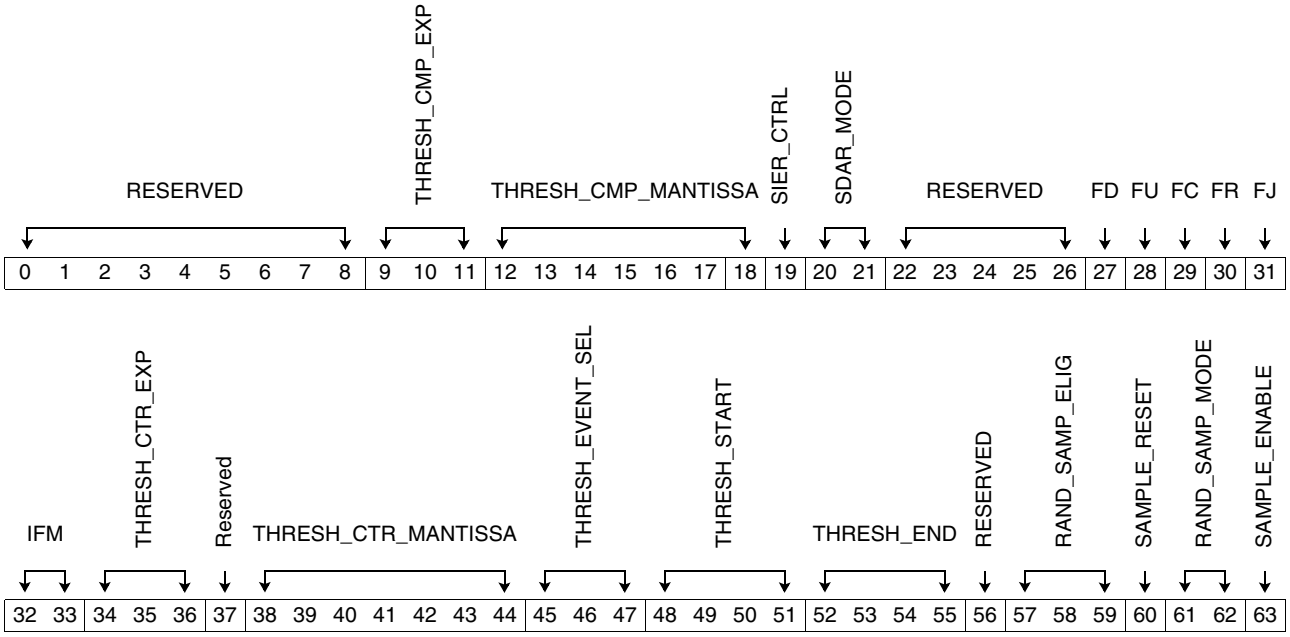
Bits	Field Name	Description
14	FC2WAIT	Freeze counter n in wait state. 0 PMCn is incremented (if permitted by other MMCR bits). 1 PMCn is not incremented if CTRL[RUN] = '0'. Software is expected to set CTRL[RUN] = '0' when it is in a wait state (for example, when there is no process ready to run). **Note:** PM_CYC counts cycles regardless of the run latch.
15	FC2H	Freeze counter n in hypervisor state. 0 PMCn is incremented (if permitted by other MMCR bits). 1 PMCn is not incremented if MSR[HV, PR] = '10'.
16	FC2TI	Freeze counter n in tags inactive mode. 0 PMCn is incremented (if permitted by other MMCR bits). 1 PMCn is not incremented in tags inactive mode.
17	FC2TA	Freeze counter n in tags active mode. 0 PMCn is incremented (if permitted by other MMCR bits). 1 PMCn is not incremented in tags active mode.
18	FC3S	Freeze counter n in privileged state. 0 PMCn is incremented (if permitted by other MMCR bits). 1 PMCn is not incremented if MSR[HV, PR] = '00'.
19	FC3P	Freeze counter n in problem state if MSR[PR] = '1'. 0 PMCn is incremented (if permitted by other MMCR bits). 1 If FCnPC = '0', PMCn is not incremented if MSR[PR] = '1'.
20	FC3PC	Freeze counter n in problem state if MSR[HV] = '1'. 0 PMCn is incremented (if permitted by other MMCR bits). 1 If FCnP = '0', PMCn is not incremented if MSR[HV, PR] = '01' (true problem state). If PCnP = '1', PMCn is not incremented if MSR[HV, PR] = '11' (adjunct state).
21	FC3M1	Freeze counter n while Mark = '1'. 0 PMCn is incremented (if permitted by other MMCR bits). 1 PMCn is not incremented if MSR[PMM] = '1'.
22	FC3M0	Freeze counter n while Mark = '0'. 0 PMCn is incremented (if permitted by other MMCR bits). 1 PMCn is not incremented if MSR[PMM] = '0'.
23	FC3WAIT	Freeze counter n in wait state. 0 PMCn is incremented (if permitted by other MMCR bits). 1 PMCn is not incremented if CTRL[RUN] = '0'. Software is expected to set CTRL[RUN] = '0' when it is in a wait state (for example, when there is no process ready to run). **Note:** PM_CYC counts cycles regardless of the run latch.
24	FC3H	Freeze counter n in hypervisor state. 0 PMCn is incremented (if permitted by other MMCR bits). 1 PMCn is not incremented if MSR[HV, PR] = '10'.
25	FC3TI	Freeze counter n in tags inactive mode. 0 PMCn is incremented (if permitted by other MMCR bits). 1 PMCn is not incremented in tags inactive mode.
26	FC3TA	Freeze counter n in tags active mode. 0 PMCn is incremented (if permitted by other MMCR bits). 1 PMCn is not incremented in tags active mode.
27	FC4S	Freeze counter n in privileged state. 0 PMCn is incremented (if permitted by other MMCR bits). 1 PMCn is not incremented if MSR[HV, PR] = '00'.

Bits	Field Name	Description
28	FC4P	Freeze counter n in problem state if MSR[PR] = '1'. 0 PMCn is incremented (if permitted by other MMCR bits). 1 If FCnPC = '0', PMCn is not incremented if MSR[PR] = '1'.
29	FC4PC	Freeze counter n in problem state if MSR[HV] = '1'. 0 PMCn is incremented (if permitted by other MMCR bits). 1 If FCnP = '0', PMCn is not incremented if MSR[HV, PR] = '01' (true problem state). If PCnP = '1', PMCn is not incremented if MSR[HV, PR] = '11' (adjunct state).
30	FC4M1	Freeze counter n while Mark = '1'. 0 PMCn is incremented (if permitted by other MMCR bits). 1 PMCn is not incremented if MSR[PMM] = '1'.
31	FC4M0	Freeze counter n while Mark = '0'. 0 PMCn is incremented (if permitted by other MMCR bits). 1 PMCn is not incremented if MSR[PMM] = '0'.
32	FC4WAIT	Freeze counter n in wait state. 0 PMCn is incremented (if permitted by other MMCR bits). 1 PMCn is not incremented if CTRL[RUN] = '0'. Software is expected to set CTRL[RUN] = '0' when it is in a wait state (for example, when there is no process ready to run). **Note:** PM_CYC counts cycles regardless of the run latch.
33	FC4H	Freeze counter n in hypervisor state. 0 PMCn is incremented (if permitted by other MMCR bits). 1 PMCn is not incremented if MSR[HV, PR] = '10'.
34	FC4TI	Freeze counter n in tags inactive mode. 0 PMCn is incremented (if permitted by other MMCR bits). 1 PMCn is not incremented in tags inactive mode.
35	FC4TA	Freeze counter n in tags active mode. 0 PMCn is incremented (if permitted by other MMCR bits). 1 PMCn is not incremented in tags active mode.
36	FC5S	Freeze counter n in privileged state. 0 PMCn is incremented (if permitted by other MMCR bits). 1 PMCn is not incremented if MSR[H,V PR] = '00'.
37	FC5P	Freeze counter n in problem state if MSR[PR] = '1'. 0 PMCn is incremented (if permitted by other MMCR bits). 1 If FCnPC = '0', PMCn is not incremented if MSR[PR] = '1'.
38	FC5PC	Freeze counter n in problem state if MSR[HV] = '1'. 0 PMCn is incremented (if permitted by other MMCR bits). 1 If FCnP = '0', PMCn is not incremented if MSR[HV, PR] = '01' (true problem state). If PCnP = '1', PMCn is not incremented if MSR[HV, PR] = '11' (adjunct state).
39	FC5M1	Freeze counter n while Mark = '1'. 0 PMCn is incremented (if permitted by other MMCR bits). 1 PMCn is not incremented if MSR[PMM] = '1'.
40	FC5M0	Freeze counter n while Mark = '0'. 0 PMCn is incremented (if permitted by other MMCR bits). 1 PMCn is not incremented if MSR[PMM] = '0'.
41	FC5WAIT	Freeze counter n in wait state. 0 PMCn is incremented (if permitted by other MMCR bits). 1 PMCn is not incremented if CTRL[RUN] = '0'. Software is expected to set CTRL[RUN] = '0' when it is in a wait state (for example, when there is no process ready to run). **Note:** PM_CYC counts cycles regardless of the run latch.

Bits	Field Name	Description
42	FC5H	Freeze counter n in hypervisor state. 0 PMCn is incremented (if permitted by other MMCR bits). 1 PMCn is not incremented if MSR[HV, PR] = '10'.
43	FC5TI	Freeze counter n in tags inactive mode.. 0 PMCn is incremented (if permitted by other MMCR bits). 1 PMCn is not incremented in tags inactive mode.
44	FC5TA	Freeze counter n in tags active mode.. 0 PMCn is incremented (if permitted by other MMCR bits). 1 PMCn is not incremented in tags active mode.
45	FC6S	Freeze counter n in privileged state. 0 PMCn is incremented (if permitted by other MMCR bits). 1 PMCn is not incremented if MSR[HV, PR] = '00'.
46	FC6P	Freeze counter n in problem state if MSR[PR] = '1'. 0 PMCn is incremented (if permitted by other MMCR bits). 1 If FCnPC = '0', PMCn is not incremented if MSR[PR] = '1'.
47	FC6PC	Freeze counter n in problem state if MSR[HV] = '1'. 0 PMCn is incremented (if permitted by other MMCR bits). 1 If FCnP = '0', PMCn is not incremented if MSR[HV, PR] = '01' (true problem state). If PCnP = '1', PMCn is not incremented if MSR[HV, PR] = '11' (adjunct state).
48	FC6M1	Freeze counter n while Mark = '1'. 0 PMCn is incremented (if permitted by other MMCR bits). 1 PMCn is not incremented if MSR[PMM] = '1'.
49	FC6M0	Freeze counter n while Mark = '0'. 0 PMCn is incremented (if permitted by other MMCR bits). 1 PMCn is not incremented if MSR[PMM] = '0'.
50	FC6WAIT	Freeze counter n in wait state. 0 PMCn is incremented (if permitted by other MMCR bits). 1 PMCn is not incremented if CTRL[RUN] = '0'. Software is expected to set CTRL[RUN] = '0' when it is in a wait state (for example, when there is no process ready to run). **Note:** PM_CYC counts cycles regardless of the run latch.
51	FC6H	Freeze counter n in hypervisor state. 0 PMCn is incremented (if permitted by other MMCR bits). 1 PMCn is not incremented if MSR[HV, PR] = '10'.
52	FC6TI	Freeze counter n in tags inactive mode. 0 PMCn is incremented (if permitted by other MMCR bits). 1 PMCn is not incremented in tags inactive mode.
53	FC6TA	Freeze counter n in tags active mode. 0 PMCn is incremented (if permitted by other MMCR bits). 1 PMCn is not incremented in tags active mode.
54:63	RESERVED	Reserved.

A.6 Monitor Mode Control Register A (MMCRA)

The 64-bit Monitor Mode Control Register A (MMCRA) gives privileged programs the ability to control the sampling process, BHRB filtering, and threshold events.



Bits	Field Name	Description	Architected
0:8	RESERVED	Reserved.	No
9:11	THRESH_CMP_EXP	The 3-bit exponent portion of threshold compare floating point.	No
12:18	THRESH_CMP_ MANTISSA	The 7-bit mantissa portion of threshold compare floating point. **Note:** The mantissa upper two bits should never be zero, unless the exponent being written is also zero. This is an artifact of the "power of 4" floating-point counter. The floating-point mantissa starts counting events starting from '000000'. When it reaches '1111111', it rolls over to '010000000' and the exponent is incremented by 1.	No
19	SIER_CTRL	When this bit is set to '1', SIER bits 16:28 are changed to read data RA bits 44:56.	No
20:21	SDAR_MODE	Continuous sampling. These bits specify how the SDAR should be updated in continuous sampling mode. 00 No updates. 01 Continuous sampling mode; update SDAR on a TLB miss. 10 Continuous sampling mode; update SDAR on a data-cache miss. 11 Continuous sampling mode; update SDAR on a store issue.	No
22:26	RESERVED	Reserved.	No
27	FD	Filter direct branch instructions. 0 Enter direct branch instructions into the BHRB if allowed by other filtering fields. 1 Do not enter direct branch instructions into the BHRB.	Yes

Bits	Field Name	Description	Architected
28	FU	Filter unconditional branch instructions. 0 Enter unconditional branch instructions into the BHRB if allowed by other filtering fields. 1 Do not enter unconditional branch instructions into the BHRB.	Yes
29	FC	Filter call instructions. 0 Enter call instructions into the BHRB if allowed by other filtering fields. 1 Do not enter call instructions into the BHRB.	Yes
30	FR	Filter return instructions. 0 Enter return instructions into the BHRB if allowed by other filtering fields. 1 Do not enter return instructions into the BHRB.	Yes
31	FJ	Filter jump instructions. 0 Enter jump instructions into the BHRB if allowed by other filtering fields. 1 Do not enter jump instructions into the BHRB.	Yes
32:33	IFM	BHRB instruction filtering mode. 00 All taken branch instructions are entered into the BHRB unless prevented by other filtering fields. 01 Do not enter jump instructions or return instructions into the BHRB.	Yes
34:36	THRESH_CTR_EXP	The 3-bit exponent portion of the threshold floating-point counter.	Yes
37	RESERVED	Reserved for architecture. This enables either the mantissa to be expanded to 8 bits or the exponent to be expanded to 4 bits. If the exponent is expanded into the reserved field, the floating-point base is changed to base 2. Changing to base 2 provides better accuracy (1.5% versus 3%), but still retains the capability of supporting very large counts.	Yes
38:44	THRESH_CTR_ MANTISSA	The 7-bit mantissa portion of threshold floating-point counter.	Yes
45:47	THRESH_EVENT_SEL	Selection for an event specified for threshold counting. PMC1 - 6 event counting, selected as shown, adheres to possible freeze conditions set up in the MMCR0, MMCR2, and MMCRC registers. 000 Do not count; disable thresholding. 001 Count the number of cycles that the CTRL run latch is set (does not depend on freeze conditions). 010 Instructions completed while the CTRL run latch is set (does not depend on freeze conditions). 011 Reserved. 100 Event configured in PMC1 (depends on freeze conditions). 101 Event configured in PMC2 (depends on freeze conditions). 110 Event configured in PMC3 (depends on freeze conditions). 111 Event configured in PMC4 (depends on freeze conditions).	Yes
48:51	THRESH_START	Threshold start event. See *Table A-4. Threshold Start/Stop Events Selection* on page 183.	Yes
52:55	THRESH_END	Threshold end event. See *Table A-4. Threshold Start/Stop Events Selection* on page 183.	Yes
56	RESERVED	Reserved.	Yes
57:59	RAND_SAMP_ELIG	Eligibility criteria. See *Table A-5. Random Sampling Eligibility Criteria* on page 184.	Yes
60	SAMPLE_RESET	When this bit is set to '1', the sampling state machine is forced from the start state into the idle state. If the state machine is not in the start state when this bit is set, it has no effect.	Yes

Bits	Field Name	Description	Architected
61:62	RAND_SAMP_MODE	Random sampling mode. 00 Random instruction sampling (RIS). Instructions that meet the criterion specified in the RAND_SAMP_ELIG field for random instruction sampling are randomly selected and sampled. 01 Random load/store sampling. Events that meet the criterion specified in the RAND_SAMP_ELIG field for random load/store sampling are randomly selected for sampling. 10 Random branch facility sampling. Events that meet the criterion specified in the RAND_SAMP_ELIG field for random branch facility sampling are randomly selected for sampling. 11 Reserved.	Yes
63	SAMPLE_ENABLE	0 Continuous sampling. 1 Random sampling.	Yes

Table A-4. lists the threshold start/stop event selection.

Table A-4. Threshold Start/Stop Events Selection

MMCRA[48] MMCRA[52]	MMCRA[49:51] MMCRA[53:55]	Description	Architected
0	000	Reserved.	Yes
0	001	Sampled instruction is decoded.	Yes
0	010	Sampled instruction is dispatched.	Yes
0	011	Sampled instruction is issued.	Yes
0	100	Sampled instruction is finished.	Yes
0	101	Sampled instruction has completed.	Yes
0	110	Sampled instruction L1 load miss.	Yes
0	111	Sampled instruction L1 reload valid.	Yes
1	000	Event selected in MMCR1 for PMC1 counting.	No
1	001	Event selected in MMCR1 for PMC2 counting.	No
1	010	Event selected in MMCR1 for PMC3 counting.	No
1	011	Event selected in MMCR1 for PMC4 counting.	No
1	100	Sampled group is next-to-complete.	No
1	101	RC machine dispatched for the sampled instruction.	No
1	110	RC machine is done for the sampled instruction.	No
1	111	Reserved.	No

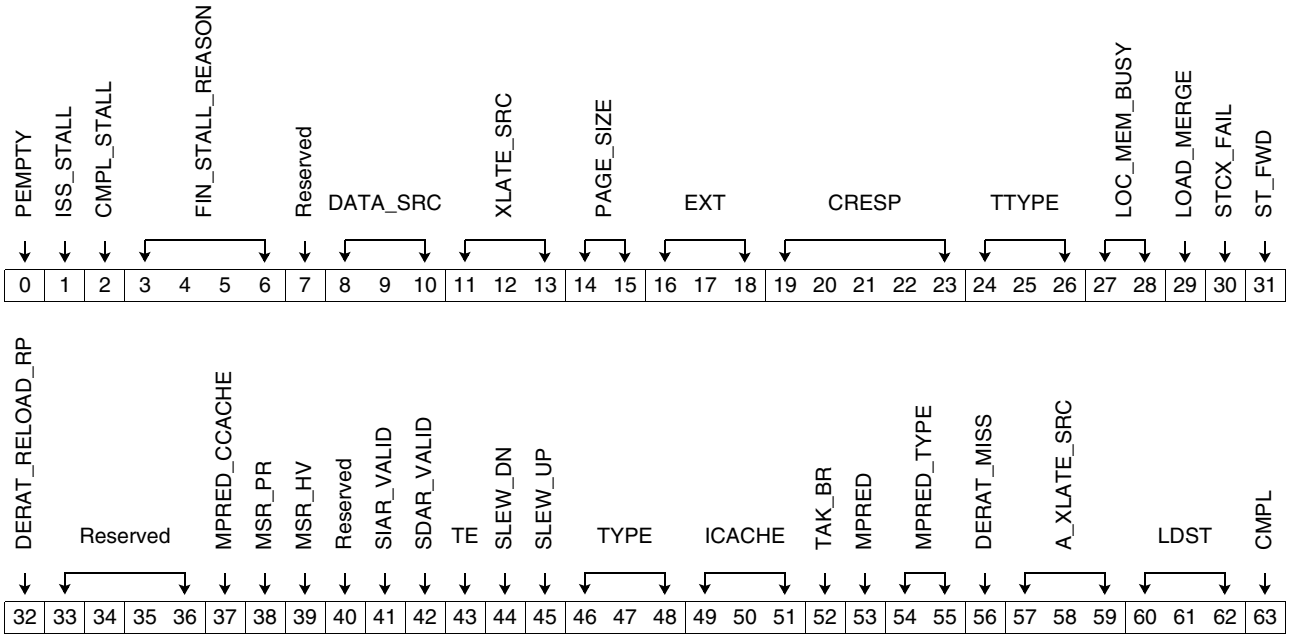
Table A-5. describes the random sampling eligibility criteria.

Table A-5. Random Sampling Eligibility Criteria

MMCRA[57:59] RAND_SAMP_ELIG	MMCRA[61:62] RAND_SAMP_MODE	MMCRA[63] SAMPLE_ENABLE	Eligibility Criteria	Architected
000	00	1	All instructions are eligible.	Yes
001	00	1	Load/store. Any operation that is routed to the LSU (for example, load or store).	Yes
010	00	1	ProbeNop.	Yes
011	00	1	Reserved.	Yes
100	00	1	IMC.	No
101	00	1	Load sampling. Sample only load instructions.	No
110	00	1	Long latency operation (**div**/**sqrt**/**mul**/**mtctr**/**br** LK = 1).	No
111	00	1	Store sampling. Sample only store instructions.	No
000	01	1	Load misses.	Yes
001	01	1	Reserved.	Yes
010	01	1	Reserved.	Yes
011	01	1	Reserved.	Yes
100	01	1	**Larx**/**stcx**	No
110	01	1	Reserved.	No
111	01	1	Reserved.	No
000	10	1	Branch mispredicts.	Yes
001	10	1	Branch mispredicts (CR).	Yes
010	10	1	Branch mispredicts (TA).	Yes
011	10	1	Taken branches.	Yes
100	10	1	Nonrepeating branches.	No
101	10	1	All branches that require prediction.	No
110	10	1	Reserved.	No
111	10	1	Reserved.	No

A.7 Sampled Instruction Event Register (SIER)

The 64-bit Sampled Instruction Event Register stores information that pertains to a sampled/marked instruction.



Bits	Field Name	Description	Architected
0	PEMPTY	Pipeline was empty (first instruction after GCT no slot).	No
1	ISS_STALL	Sampled instruction issue stall. The marked instruction was issued after it became NTC.	No
2	CMPL_STALL	Sampled instruction group completion stall. The marked instruction became NTC some time before completion.	No
3:6	FIN_STALL_REASON	Sampled instruction group finish stall reason. 0000 Reserved. 0001 Marked finish before NTC 0010 Reserved 0011 Reserved 0100 LSU D-cache miss 0101 LSU load finish 0110 LSU store forward 0111 LSU store 1000 FXU multi-cycle 1001 BRU finish mispredict 1010 VSU multi-cycle 1011 Reserved 1100 FXU other 1101 VSU other 1110 BRU other 1111 Reserved	No
7	Reserved	Reserved.	No

Bits	Field Name	Description	Architected
8:10	DATA_SRC	Sampled instruction microarchitecture-dependent data source information for loads and extended store information. (Encoding defined in *Table A-6.* on page 188.)	No
11:13	XLATE_SRC	Sampled instruction microarchitecture-dependent data source information included with data translation source load information. (Encoding defined in *Table A-6.* on page 188.)	No
14:15	PAGE_SIZE	Sampled instruction suffered an ERAT miss for the page size. 00 4 KB 01 64 KB 10 16 MB 11 16 GB	No
16:18	EXT	Sampled Instruction data/store/translation extension information. (Encoding defined in *Table A-7.* on page 190.)	No
19:23	CRESP	Combined response.	No
24:26	TTYPE	SMP interconnect event ttype.	No
27:28	LOC_MEM_BUSY	Local memory busy.	No
29	LOAD_MERGE	Load merge.	No
30	STCX_FAIL	Sampled **stcx** failed.	No
31	ST_FWD	Store forward.	No
32	DERAT_RELOAD_RP	Sampled instruction suffered a DERAT miss and reloaded a Page Table Entry (PTE) in Radix Page Translation (RPT) mode.	No
33:36	Reserved	Reserved.	No
37	MPRED_CCACHE	C-cache misprediction.	No
38	MSR_PR	Sampled problem state (MSR[PR] = '1').	Yes
39	MSR_HV	Sampled hypervisor state (MSR[HV] = '1').	Yes
40	Reserved	Reserved	No
41	SIAR_VALID	SIAR is valid for sampled instruction.	Yes
42	SDAR_VALID	SDAR is valid for sampled instruction.	Yes
43	TE	Threshold exceeded.	Yes
44	SLEW_DN	Frequency was slewed down for any reason.	Yes
45	SLEW_UP	Frequency was slewed up for any reason.	Yes
46:48	TYPE	Type of sampled instruction. 000 Reserved. 001 Sampled instruction is a load. 010 Sampled instruction is a store. 011 Sampled instruction is a branch. 100 Sampled instruction is a floating-point instruction. 101 Sampled instruction is a fixed-point instruction. 110 Sampled instruction is an IFU but nonbranch. 111 Reserved.	Yes

Bits	Field Name	Description	Architected
49:51	ICACHE	000 Reserved 001 Sampled instruction hit in the I-cache 010 Sampled instruction hit in the L2 cache. 011 Sampled instruction hit in the L3 cache. 100 Sampled instruction L3 miss. 101 Reserved. 110 Reserved. 111 Reserved. **Note:** This field is not valid when random event sampling is enabled MMCRA[63] = '1' and MMCRA[61:62] ≠ '00'.	Yes
52	TAK_BR	Sampled instruction is a taken branch.	Yes
53	MPRED	Sampled branch instruction is mispredicted.	Yes
54:55	MPRED_TYPE	Sampled instruction branch mispredicted information type. 00 Reserved. 01 Branch mispredicted due to direction. 10 Branch mispredicted due to target address. 11 Reserved.	Yes
56	DERAT_MISS	Sampled instruction incurred a D-ERAT miss.	Yes
57:59	A_XLATE_SRC	Sampled instruction data translation information. 000 Reserved. 001 TLB hit. 010 Data translation hit in the L2 cache. 011 Data translation hit in the L3 cache. 100 Data translation resolved from memory. 101 Data translation resolved from on-chip cache (other than local L2/L3 cache). 110 Data translation resolved from off-chip cache. 111 Reserved.	Yes
60:62	LDST	Sampled load/store instruction information. 000 Reserved. 001 Load hit the L1 D-cache. 010 Load hit in the L2 cache. 011 Load hit in the L3 cache. 100 Load resolved from memory. 101 Load resolved from on-chip cache (other than local L2/L3 cache). 110 Load resolved from off-chip cache. 111 Store missed L1 and sent to cache/memory subsystem.	Yes
63	CMPL	Sampled instruction completed.	Yes

Table A-6. shows the implementation-dependent extension to data source encodes.

Table A-6. Implementation-Dependent Extension to Data Source Encodes (Sheet 1 of 2)

Architected Bits SIER[LDST] SIER[A_XLATE_SRC]	Implementation-Dependent Bits Encoding SIER[DATA_SRC] SIER[XLATE_SRC]	Performance Monitor Reload Bus Source Encoding (Binary)	Implementation-Dependent Bit Description
L2 Hit			
010	000	00000	Private L2 cache for this core-sourced data (or NCU loads) is without dispatch conflicts.
010	001	00001	Private L2 cache for this core-sourced data in the Mepf state is without dispatch conflicts (L3 prefetch brought line in Me).
010	010	00010	Private L2 cache for this core-sourced data that had a dispatch conflict on load-hit-store.
010	011	00011	Private L2 cache for this core-sourced data that had a dispatch conflict other than a load-hit-store.
010	100		Reserved.
010	101		Reserved.
010	110		Reserved.
010	111		Reserved.
L3 Hit			
011	000	00100	Private L3 cache for this core-sourced data without dispatch conflicts.
011	001	00101	Private L3 cache for this core-sourced data in the Mepf state without dispatch conflicts (L3 prefetch brought line in Me).
011	010	00111	Private L3 cache for this core-sourced data that had a dispatch conflict.
011	011		Reserved.
011	100		Reserved.
011	101		Reserved.
011	110		Reserved.
011	111		Reserved.
Memory			
100	000	11010	Cacheable load on chip from the L4 cache.
100	001	11011	Cacheable load on chip from memory.
100	010	11100	Cacheable load within six chips from the L4 cache.
100	011	11101	Cacheable load within six chips from memory.
100	100	11110	Cacheable load beyond six chips from the L4 cache.
100	101	11111	Cacheable load beyond six chips from memory.
100	110		Reserved.
100	111		Reserved.

Table A-6. Implementation-Dependent Extension to Data Source Encodes (Sheet 2 of 2)

Architected Bits SIER[LDST] SIER[A_XLATE_SRC]	Implementation-Dependent Bits Encoding SIER[DATA_SRC] SIER[XLATE_SRC]	Performance Monitor Reload Bus Source Encoding (Binary)	Implementation-Dependent Bit Description
On-Chip Cache			
101	000	11000	Data sourced from another L2.1 cache (not M but valid) on the same chip.
101	001	11001	Data sourced from another L2.1 cache (M) on the same chip.
101	010	10000	Data sourced from another L3.1 cache (not M but valid) on the same chip.
101	011	10001	Data sourced from another L3.1 cache (M) on the same chip
101	100	10010	Data sourced from another L3.1 cache (not M but valid) on the same chip (ECO).
101	101	10011	Data sourced from another L3.1 cache (M) on same chip (ECO).
101	110		Reserved
101	111		Reserved
Off-Chip Cache			
110	000	10100	Data sourced from another L2 or L3 cache (not M but valid) within six chips.
110	001	10101	Data sourced from another L2 or L3 cache (M) within six chips.
110	010	10110	Data sourced from another L2 or L3 cache (not M but valid) beyond six chips.
110	011	10111	Data sourced from another L2 or L3 cache (M) beyond six chips.
110	100		Reserved
110	101		Reserved
110	110		Reserved
110	111		Reserved
Store Information (only applicable for SIER[LDST])			
111	000		Store did not require an RC dispatch.
111	001		Store completed in the L2 cache without intervention.
111	010		Store completed in the L2 cache with modified intervention.
111	011		Reserved
111	100		Reserved
111	101		Reserved
111	110		Reserved
111	111		Reserved

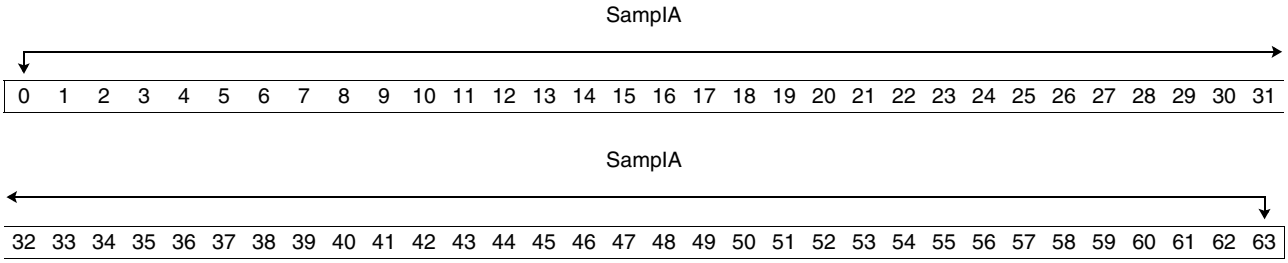
Table A-7. shows the implementation-dependent extension bits for data source encodes (SIER[EXT]).

Table A-7. Implementation-Dependent Extension Bits for Data Source Encodes (SIER[EXT])

Encoding	Case	Description	Prediction
000	Nonfabric operation	Private L2/L3 hit for this core-sourced data (or NCU load).	NA
001	Fabric Initial/Final Pump = Chip	Initial and final pump scope and data sourced across this scope.	Correct
010	Fabric Initial/Final Pump = Group	Initial and final pump scope and data sourced across this scope.	Correct
011	Fabric Initial/Final Pump = System	Initial and final pump scope and data sourced across this scope.	Correct
100	Fabric Final Pump = Group	Final pump scope for data sourced finished larger than the initial pump scope (started too small).	Mispredict
101	Fabric Final Pump = Group	Final pump scope received data from a source that was at a smaller scope (should have started smaller).	Mispredict
110	Fabric Final Pump = System	Final pump scope to receive data sourced ended with larger than initial pump scope (started too small).	Mispredict
111	Fabric Final Pump = System	Final pump scope received data from source that was at a smaller scope.	Mispredict

A.8 Sampled Instruction Address Register (SIAR)

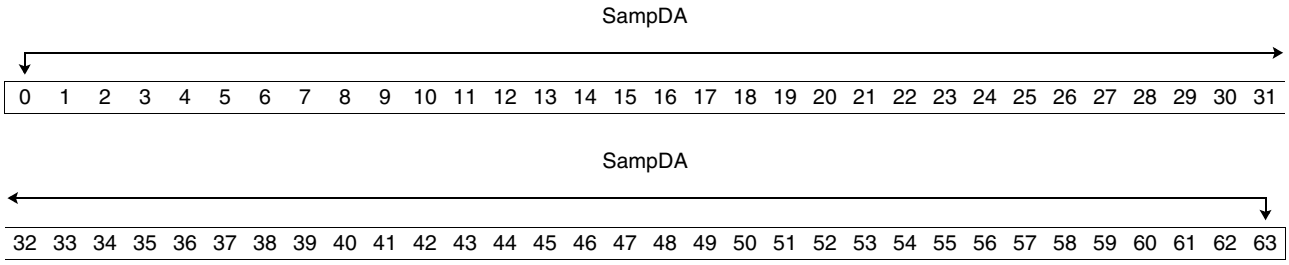
The 64-bit Sampled Instruction Address Register contains the instruction address relating to a sampled/marked instruction. The PMU related bits in the SIAR Register are shown in the following bit diagram.



Bits	Field Name	Description
0:63	SampIA	Sampled instruction address.

A.9 Sampled Data Address Register (SDAR)

The 64-bit Sampled Data Address Register contains the data address relating to a sampled/marked instruction. PMU-related bits in the SDAR Register are shown in the following bit diagram. A counter-negative condition exists when the value in a PMC is negative.



Bits	Field Name	Description
0:63	SampDA	Sampled data address.

Glossary

ALU	Arithmetic logic unit
AMO	Atomic memory operation
APC	Application/acceleration processing
apcfsm	APC finite state machine
AS	Accelerator switchboard
BCD	Binary-coded decimal
BESCR	Branch Event Status and Control Register
BFU	Binary floating-point unit
BHRB	Branch history rolling buffer
BHT	Branch history table
BRQ	Branch queue
BRU	Branch register unit
BTAC	Branch target address cache
CAM	Content addressable memory
CAPI	Coherently attached processor interface
CAPP	Coherent accelerator processor proxy
CAS	Column address strobe
C-cache	Count cache
CDF	Critical data forward
CI	Castin
CLB	Control logic block
clrbhrb	Clear branch history rolling buffer
CNPM	Centralized nest performance monitor
CO	Castout
CPI	Cycles per instruction
CPU	Central processing unit
CRU	Condition Register unit
DARQ	Data and address recirculation queue
DDR	Double data rate
D-ERAT	Data effective-to-real-address translation
DFU	Decimal floating-point unit or quad-precision floating-point unit

DIMM	Dual in-line memory module
DMA	Direct memory attach
DNPM	Distributed nest performance monitor
DP	Double precision
DPTEG	Data page table entry group
DRAM	Dynamic random-access memory.
DTS	Direct-tree source
EAT	Effective address table
ECC	Error correction code
ECO	Extended cache option
EMQ	ERAT miss queue
ERAT	Effective-to-real-address translation, or a buffer or table that contains such translations, or a table entry that contains such a translation.
FLOP	Floating-point operation
FMA	Floating multiply add instruction
FPGA	Field-programmable gate array
FPSCR	Floating-Point Status and Control Register
FSCR	Facility Status and Control Register
FXU	Fixed-point unit
GCT	Global completion table
GS	SMP interconnect group scope request
HFSCR	Hypervisor Facility Status and Control Register
HPC_WR	Highest point of coherency write
HPT	Hashed page table
I^2C	Inter-integrated circuit
IAR	Instruction Address Register
ICT	Instruction completion table
IFU	Instruction fetch unit
IMC	Instruction Match CAM or in-memory collection
iop	Internal operation
IPTEG	Instruction page table entry group
ISA	Instruction set architecture
ISU	Instruction dispatch and issue unit

ITLB	Instruction translation lookaside buffer
Ix	I-state (invalid)
L1	Level 1
L2	Level 2
L3	Level 3
L4	Level 4
LCO	Lateral castout
LHS	Load-hit-store
LMQ	Load-miss queue
LNS	SMP interconnect local node scope request
LPAR	Logical partition
LRQ	Load reorder queue
LSAQ	Load-store address queue
LSU	Load store unit
MBA	Memory buffer asynchronous
MC	Memory controller
MCD	Memory coherency directory
MCS	Memory controller synchronous
mfbhrb	Move from branch history rolling buffer
mfmsr	Move from Machine State Register instruction
mfspr	Move from special-purpose register instruction
MMCR	Monitor Mode Control Register
MMIO	Memory-mapped input/output
MMU	Memory management unit
Mpred	Misprediction
Msb	Most-significant bit
mtmsr	Move to Machine State Register instruction
mtspr	Move to special-purpose register instruction
NCU	Non-cacheable unit
NMMU	Nest memory management unit
NNS	Near node scope
NOP	No operation

NPU	NVLink processing unit
NTC	Next instruction to complete
OW	Octword
PCI	Peripheral Component Interconnect
PCIe	Peripheral Component Interconnect Express
PCP	Performance co-pilot
PDE	Page directory entry
PEC	PCI Express controller
PHB	PCIe host bridge
PMC	Performance monitor counter
PMCD	Performance monitor counter daemon
PMDA	Performance metric domain agent
PMU	Performance monitor unit
PSL	Power service layer
PTE	Page table entry
PRTE	Process table entry
PURR	Processor Utilization Resource Register
RA	Read address
RC	Read claim
RDR	L2 to L3 read request bus
RES	Random event sampling
RFID	Return from interrupt data
RIS	Random instruction sampling
RNS	Remote node scope
ROT	Rollback-only transaction
rty	Retry
rwitm	Read with intent to modify
SAO	Strong address ordering
SC	Store clean
SDAR	Sampled Data Address Register
SIAR	Sample Instruction Address Register
SIER	Sample Instruction Event Register

SLB	Segment look-aside buffer
SMP	Symmetric multiprocessor
SMT	Simultaneous multithreading
SN	L2 Snoop machine
SNP	L2 Snoop request from the SMP interconnect
SPR	Special purpose register
SPURR	Scaled Processor Utilization Resource Register
SQ	L2 store queue
SRQ	Store reorder queue
ST	Single thread
Sx	S-state (shared)
TEXASR	Transaction Exception And Summary Register
TLB	Translation lookaside buffer
TM	Transactional memory
Tx	T-state (tagged)
uOP	Micro-operation
VAS	Virtual accelerator switchboard
VDP	Vector double-precision
VFX	Vector fixed-point unit
VGS	SMP interconnect vector group scope request
VHDL	VHSIC Hardware Description Language
VHSIC	Very high-speed integrated circuit
VSU	Vector scalar unit
WI	Write inject
WOF	Workload-optimized frequency
XIVE	External interrupt virtualizer engine