[Public]

AMD64 Technology Platform Quality of Service Extensions

Publication # **56375** Revision: **1.03**
Issue Date: **February 2022**

Advanced Micro Devices

The information contained herein is for informational purposes only, and is subject to change without notice. While every precaution has been taken in the preparation of this document, it may contain technical inaccuracies, omissions and typographical errors, and AMD is under no obligation to update or otherwise correct this information. Advanced Micro Devices, Inc. makes no representations or warranties with respect to the accuracy or completeness of the contents of this document, and assumes no liability of any kind, including the implied warranties of noninfringement, merchantability or fitness for particular purposes, with respect to the operation or use of AMD hardware, software or other products described herein. No license, including implied or arising by estoppel, to any intellectual property rights is granted by this document. Terms and limitations applicable to the purchase or use of AMD's products are as set forth in a signed agreement between the parties or in AMD's Standard Terms and Conditions of Sale.

Trademarks

AMD, the AMD Arrow logo, and combinations thereof, are trademarks of Advanced Micro Devices, Inc. Other product names used in this publication are for identification purposes only and may be trademarks of their respective companies.

Contents

Revision History

Date	Rev.	Description
January 2022	1.03	• Added definition of Bandwidth Monitoring Event Configuration (BMEC) • L3 External Slow Memory Bandwidth Allocation Enforcement (L3SBE)
October 2020	1.02	• Updated Introduction. • Added table "Platform Quality of Service Extension Versions". • Added table "Counter Size for PQoS Versions". • Added table "CPUID Fn0000_000F_EAX_x1 Platform QOS Monitor Capability (ECX=1)". • Updated Platform QOS Monitoring of L3 External Bandwidth section. Added table "EvtID Used for Request Types". • Updated Code and Data Prioritization (CDP) section. • Updated Platform QOS Limits for Bandwidth Enforcement section.
September 2018	1.01	• Updated page 5 first paragraph in Presence and Capabilities section, page 12 first paragraph, and page 17 first paragraph. • Updated QM_CTR, MSR C8Eh table on page 11.
August 2018	1.00	Initial public release.

Introduction

This document describes proposed AMD64 architectural extensions for Platform Quality of Service. These extensions are intended to provide for the *monitoring* of the usage of certain system resources by one or more processors and for the separate allocation and *enforcement* of limits on the use of certain system resources by one or more processors. The monitoring and enforcement are not necessarily applied across the entire system, but in general apply to a QOS domain which corresponds to some shared system resource. The QOS domain is a Core-Complex in some implementations. The set of resources which are monitored and the set for which the enforcement of limits is provided are implementation dependent.

Platform QOS features are implemented on a logical processor basis. Therefore, multiple hardware threads of a single physical CPU core may have independent resource monitoring and enforcement configurations. For the rest of this document, the term “processor” should be taken to mean a “logical processor”; that is, potentially one thread of a multi-threaded processor.

Presence and Capabilities

Detection of the presence of these extensions and their attributes is performed with the following CPUID functions. Use the following table to interpret CPUID functions that refer to the Family/Stepping of a product. This table identifies the Platform Quality of Service Extension Version (PQoS Version) associated with a product of a specific Family/Model.

Platform Quality of Service Extension Versions

Family	Model	PQoS Version
17h	30h–9Fh	V1.0
19h	00h–0Fh	V2.0
19h	20h–5Fh	V2.0

CPUID function 7 Indicates the overall support for Platform QOS Monitoring (PQM, EBX bit 12) and Platform QOS Enforcement (PQE, bit 15).

CPUID Fn0000_0007_EBX_x0 Structured Extended Feature Identifiers (ECX=0)

Bits	Field Name	Description
15	PQE	Platform QOS Enforcement
12	PQM	Platform QOS Monitoring

Once overall support for PQM has been established, support for specific sub-features may be determined using CPUID function 0xF and 0x8000_0020.

In CPUID function 0xF, sub-function ECX = 0 returns a bit vector in EDX identifying the sub-features supported. EBX returns the maximum RMID number supported by any PQM feature.

CPUID Fn0000_000F_EDX_x0 Platform QOS Monitor Capability (ECX=0)

Bits	Field Name	Description
1	L3CacheMon	L3 Cache Monitoring support

CPUID Fn0000_000F_EBX_x0 Platform QOS Monitor Capability (ECX=0)

Bits	Field Name	Description
31:0	MAX_RMID	Largest RMID supported by the system for any resource

Specific attributes associated with a given PQM sub-feature may be obtained through CPUID function 0xF with ECX equal to the sub-feature number. For L3 Cache Monitoring, these attributes are as follows: ECX returns the maximum RMID number for L3 Cache Monitoring. EDX bit 0 indicates support for L3 Occupancy Monitoring, EDX bit 1 indicates support for monitoring the total bandwidth supplied to the L3 from the rest of the system (lower levels of the cache hierarchy if present, and memory) and EDX bit 2 indicates support for monitoring the bandwidth supplied to the L3 from the local NUMA domain. EBX returns the L3 Cache Conversion Factor. To obtain the cache occupancy in bytes (or the bandwidth in bytes for bandwidth measurements, specified below), the value returned from the QM_CTR is multiplied by this conversion factor.

EAX returns the size of the QOS Bandwidth Monitoring counters. If CounterSize is non-zero, the value of CounterSize + 24 is the number of bits in the counter. For example, if the value of CounterSize is 20, then the QOS Bandwidth Monitoring Counters are 44 bits wide. If CounterSize is 0, use the PQoS Version identified by the product Family/Model at the beginning of the document and the following table to identify the counter size.

Counter Size for PQoS Versions

PQoS Version	QOS Bandwidth Monitoring Counter Size (Bits)
V1.0	62
V2.0	44

CPUID Fn0000_000F_ECX_x1 Platform QOS Monitor Capability (ECX=1)

Bits	Field Name	Description
31:0	MAX_RMID	Largest RMID supported for this resource

CPUID Fn0000_000F_EDX_x1 Platform QOS Monitor Capability (ECX=1)

Bits	Field Name	Description
2	L3CacheLclBWMon	Monitoring of L3 Cache bandwidth from rest of system, local
1	L3CacheTotBWMon	Monitoring of L3 Cache bandwidth from rest of system, total
0	L3CacheOccMon	L3 Cache occupancy monitoring

CPUID Fn0000_000F_EBX_x1 Platform QOS Monitor Capability (ECX=1)

Bits	Field Name	Description
31:0	ConvertFactor	L3 Cache Conversion Factor

CPUID Fn0000_000F_EAX_x1 Platform QOS Monitor Capability (ECX=1)

Bits	Field Name	Description
8	OverflowBit	Indicates QM_CTR bit 61 is an overflow bit
7:0	CounterSize	QOS Bandwidth Monitoring Counter Size offset from bit 24. If value is 0, refer to the table above to identify the counter size based on the Family/Stepping.

In CPUID function 0x8000_0020, sub-function ECX=0 returns a bit vector in EBX identifying additional PQM sub-features. The remaining bits in the CPUID identify PQE sub-features or are Reserved.

CPUID Fn8000_0020_EBX_x0 Platform QOS Extended Feature Identifiers – PQM Features (ECX=0)

Bits	Field Name	Description
3	EVT_CFG	Bandwidth Monitoring Event Configuration (BMEC)

Specific attributes associated with a given PQM Extended sub-feature may be obtained through CPUID function 0x8000_0020 with ECX equal to the sub-feature number.

For EVT_CFG, which identifies the prescence of Bandwidth Monitoring Event Configuration (BMEC), EBX identifies the number of Configurable Events available to the user, and ECX identifies the Bandswidth Types available to BMEC.

CPUID Fn8000_0020_EBX_x3 Bandwidth Monitoring Event Configuration (ECX=3)

Bits	Field Name	Description
7:0	EVT_NUM	The number of bandwidth events available to be configured

CPUID Fn8000_0020_EBX_x3 Bandwidth Monitoring Event Configuration (ECX=3)

Bits	Field Name	Description
6	L3CacheBwVicMon	Dirty Victims from the QOS domain to all types of memory
5	L3CacheRmtSlowBwFillMon	Reads to slow memory in the non-local NUMA domain
4	L3CacheLclSlowBwFillMon	Reads to slow memory in the local NUMA domain
3	L3CacheRmtBwNtWrMon	Non-temporal writes to non-local NUMA domain
2	L3CacheLclBwNtWrMon	Non-temporal writes to local NUMA domain
1	L3CacheRmtBwFillMon	Reads to memory in the non-local NUMA domain
0	L3CacheLclBwFillMon	Reads to memory in the local NUMA domain

When PQE support is indicated by CPUID function 0x7, EBX bit 15, support for Platform Quality of Service Enforcement sub-features may be determined using CPUID function 0x10 with ECX = 0 and by using CPUID function 0x8000_0008 with ECX = 0.

CPUID function 0x10 with ECX = 0 returns a bit vector of supported sub-features related to cache capacity allocation enforcement in EBX. Currently the only defined sub-feature is L3 cache allocation enforcement, reported in EBX bit 1.

CPUID Fn0000_0010_EBX_x0 Platform QOS Enforcement Capability (ECX=0)

Bits	Field Name	Description
1	L3Alloc	L3 Cache allocation enforcement

When support for L3 cache allocation enforcement has been determined as above, CPUID function 0x10 with ECX = 1 (the sub-function number) may be used to determine the attributes of the L3 cache allocation enforcement feature. EAX bits [4:0] return the size of the L3 capacity allocation bitmask, minus 1. That is, a return value of 0xF would indicate a 16-bit bitmask ([15:0]). EBX bits [31:0] indicate the portion of the L3 cache which may be shared with other system entities not under Platform QOS Enforcement control. ECX bit 2 indicates support for Code and Data Prioritization for L3 cache allocations. EDX bits [15:0] return the highest COS number which the system supports.

CPUID Fn0000_0010_EAX_x1 L3 Cache Allocation Enforcement (ECX=1)

Bits	Field Name	Description
4:0	CBM_LEN	L3 Cache capacity bitmask length, zero based

CPUID Fn0000_0010_EBX_x1 L3 Cache Allocation Enforcement (ECX=1)

Bits	Field Name	Description
31:0	L3ShareAllocMask	L3 Cache allocation sharing mask

CPUID Fn0000_0010_ECX_x1 L3 Cache Allocation Enforcement (ECX=1)

Bits	Field Name	Description
2	CDP	Code-Data Prioritization support

CPUID Fn0000_0010_EDX_x1 L3 Cache Allocation Enforcement (ECX=1)

Bits	Field Name	Description
15:0	COS_MAX	Maximum COS for L3 Cache allocation enforcement

When PQE support is indicated by CPUID function 0x7, EBX bit 15, CPUID function 0x8000_0008 with ECX = 0 will return an indication in EBX bit 6 that AMD Bandwidth Enforcement is supported.

CPUID Fn8000_0008_EBX Extended Feature Identifiers

Bits	Field Name	Description
6	BE	AMD Bandwidth Enforcement

The sub-features supported under AMD Bandwidth Enforcement are identified using CPUID function 0x8000_0020, ECX = 0. This function returns a bit vector of supported sub-features related to bandwidth allocation enforcement in EBX. Currently the defined PQE sub-feature are L3 external bandwidth allocation enforcement (L3BE), reported in EBX bit 1; and L3 External Slow Memory Bandwidth Allocation Enforcement (L3SBE), reported in EBX bit 2.

CPUID Fn8000_0020_EBX_x0 AMD Bandwidth Enforcement Feature Identifiers (ECX=0)

Bits	Field Name	Description
2	L3SBE	L3 external slow memory bandwidth enforcement
1	L3BE	L3 external bandwidth enforcement

Specific attributes associated with a given PQM extended sub-feature may be obtained through CPUID function 0x8000_0020 with ECX equal to the sub-feature number.

The attributes associated with L3 external bandwidth enforcement are determined using CPUID function 0x8000_0020, ECX = 1 (the sub-feature number). EAX returns the number of bits available to specify the bandwidth limit to be enforced. EDX returns the maximum COS number available for L3 external bandwidth enforcement.

CPUID Fn8000_0020_EAX_x1 L3 External Bandwidth Enforcement (ECX=1)

Bits	Field Name	Description
31:0	BW_LEN	L3 External Bandwidth Enforcement bit range length

CPUID Fn8000_0020_EDX_x1 L3 External Bandwidth Enforcement (ECX=1)

Bits	Field Name	Description
31:0	COS_MAX	Maximum COS number for L3 External Bandwidth Enforcement

The attributes associated with L3 External Slow Memory Bandwidth Enforcement are determined using CPUID fucntion 0x8000_0020, ECX=2. EAX returns the number of bits available to specify the bandwidth limit to be enforced. EDX returns the maximum COS number available for L3 External Slow Memory Bandwidth Enforcement.

CPUID Fn8000_0020_EAX_x2 External Slow Memory Bandwidth Enforcement (ECX=2)

Bits	Field Name	Description
31:0	BW_LEN	L3 External Slow Memory Bandwidth Enforcement Bit Range Length

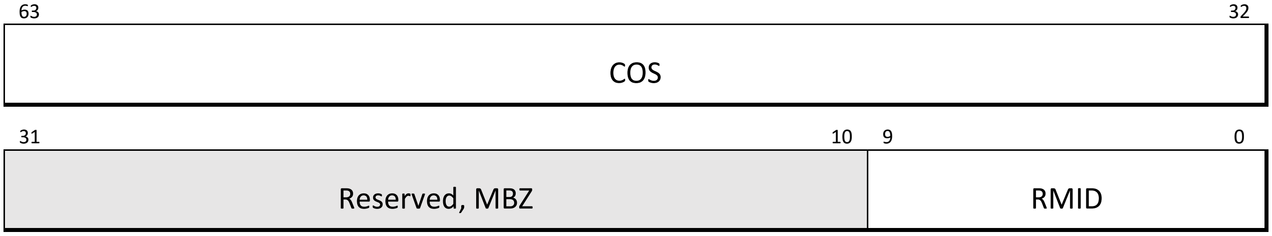
CPUID Fn8000_0020_EDX_x2 External Slow Memory Bandwidth Enforcement (ECX=2)

Bits	Field Name	Description
31:0	COS_MAX	Maximum COS number for L3 External Slow Memory Bandwidth Enforcement

Usage

QOS Monitoring

QOS monitoring is accomplished by assigning a Resource Management Identifier (RMID) to a given processor or group of processors and at some later time reading the usage of a particular monitored resource. The association of an RMID with a processor is performed by writing the desired RMID number into PQR_ASSOC (MSR C8Fh). At reset, the RMID value is 0.



Bits	Mnemonic	Description	R/W
63:32	COS	Class of Service	R/W
31:10	Reserved	Reserved, Must be Zero	
9:0	RMID	Resource Monitor Identifier	R/W

PQR_ASSOC, MSR C8Fh

The COS field of PQR_ASSOC is used for QOS enforcement, discussed below.

Reading the usage of a particular resource is accomplished by writing the RMID and desired event ID to the QM_EVTSEL (MSR C8Dh) register and then reading the usage count from the QM_CTR (MSR C8Eh) register. The event ID is one more than the feature number as reported in EDX by CPUID function 0xF, ECX=1. For example, the event number for L3 Cache Occupancy Monitoring is 0, so the event ID is 1. Writing to any of the Reserved bits in the QM_EVTSEL register or writing a value larger than the maximum RMID as returned in EBX by CPUID function 0xF with ECX=0 will result in a #GP. Writing an undefined Event ID or an RMID value greater than the largest RMID supported for that specific resource to the QM_EVTSEL register will cause the subsequent read of the QM_CTR to return “1” in the “E” (error) field.

The hardware may return a “1” in the “U” position (bit 62) of the QM_CTR at any time. This indicates that the hardware was unable to return the requested event count for the specified RMID at that time. This should generally be a temporary condition and subsequent reads may succeed.

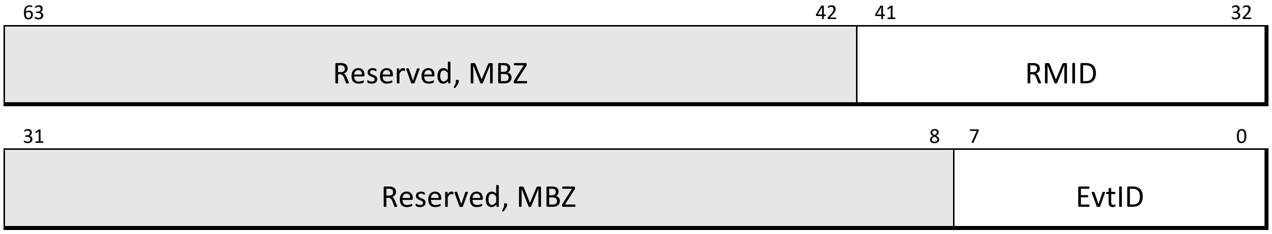
Software may associate multiple processors with the same RMID at the same time and, if the processors are located within the same QOS domain, the total resource usage by all of these processors will be accumulated together and the total reported by the hardware.

For PQoS Version 1.0 and 2.0, as identified by Family/Model, the QM_EVTSEL register is shared by all the processors in a QOS domain. It is software’s responsibility to ensure that no other process changes the QM_EVTSEL register between the time it is written and the time the QM_CTR read occurs. For other implementations, each processor in the QOS domain has a private QM_EVTSEL and QM_CTR register pair, so software does not require synchronization between processors in the QOS domain to use the register pair.

The PQR_CTR register is not writable. Attempts to write it will cause a #GP.

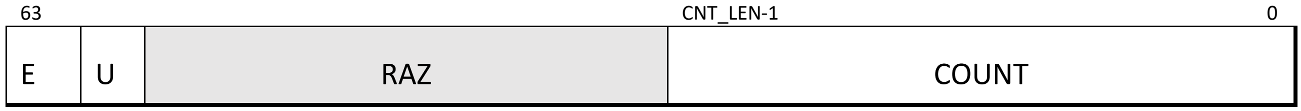
For “rate” events which continually increment the count (such as bandwidth), when the maximum count is reached, the counter will roll over and continue counting. It is software’s responsibility to read the count often enough to avoid having the count roll over twice between reads. The width of the COUNT field of the PQR_CTR register may vary between implementations and is enumerated in CPUID Fn0000_000F_EAX_x1. All implementations will support a CNT_LEN sufficient to ensure that the count will not roll over in less than one second.

Specific PQM events are described below.



Bits	Mnemonic	Description	R/W
63:42	Reserved	Reserved, Must be Zero	
41:32	RMID	Resource Monitor Identifier	R/W
31:8	Reserved	Reserved, Must be Zero	
7:0	EvtID	PQM Event Identifier	R/W

QM_EVTSEL, MSR C8Dh



Bits	Mnemonic	Description	R/W
63	E	Error on access to the counter; illegal event or RMID	R
62	U	Count for this event is currently not available for RMID	R
61:CNT_LEN	RAZ	Returns 0 when read	
CNT_LEN-1:0	COUNT	Count of monitored resource event	R

QM_CTR, MSR C8Eh

Platform QOS Monitoring of L3 Occupancy

The QOS domain for L3 occupancy monitoring is an L3 cache and all the processors which share that cache. When using L3 occupancy monitoring, L3 cache lines are associated with the current RMID value in the PQR_ASSOC register of the processor which accessed the cache line. When the QM_CTR is read for that RMID, the hardware reports an *approximation* of the amount of the L3 cache contents currently associated with that RMID in the L3 QOS domain of the processor performing the read. The number returned should be multiplied by the L3 Cache Conversion Factor to obtain the approximate number of bytes in the L3 cache currently associated with the supplied RMID. The L3 Cache Conversion Factor is returned in EBX by the CPUID function 0xF, ECX=1.

If two processors in the same QOS domain are running with different RMIDs and both processors access a given cache line, the line may be associated with either of the RMIDs. Subsequently, the line may remain associated with the RMID with which it was first associated, or its RMID association may change over time, depending on access patterns.

Similarly, if the RMID of a processor is changed but that processor continues to access cache lines which have been associated with the previous RMID, the relevant cache lines may continue to be reported as belonging to the first RMID for an indefinite period.

If an RMID has been in use by one or more processors within a given QOS domain and software changes the associations of RMIDs with processors so that no processors remain associated with that RMID, the L3 occupancy count for that RMID will not immediately drop to zero. This is because lines which were installed in the L3 cache and associated with that RMID may remain in the cache until they are replaced through normal cache replacement mechanisms or software explicitly flushes the lines with CLFLUSH or WBINVD instructions.

Note: *Hardware may not keep a precise account of the cache occupancy associated with a given RMID so software must not assume that an L3 occupancy measurement of zero for a given RMID is an indication that no lines remain in the cache which were installed by processors running with the given RMID. That is, the L3 cache occupancy monitor may never be used to determine whether cache flushing may be skipped in situations which would ordinarily require cache flushing for functional correctness.*

Platform QOS Monitoring of L3 External Bandwidth

Whenever an RMID value is written to the PQR_ASSOC MSR of a processor, the hardware will begin counting requests issued by the L3 cache to the rest of the system, not including I/O (that is, to lower level caches or memory).

The types of system requests counted by the hardware depends on the implementation. If BMEC is not implemented (as determined by CPUID), requestes counted are defined by the table below. Each cell specifies which EvtID a request type is counted as in various implementations identified by the PQoS Version identified by the Family/Model of the product.

EvtID Used for Request Types

Request Type\PQoS Version	V1.0	V2.0
Read to Local NUMA Domain	2, 3	2, 3
Read to Non-Local NUMA Domain	2	2
Non-Temporal Write to Local NUMA Domain	N/A	2, 3
Non-Temporal Write to Non-Local NUMA Domain	N/A	2

Therefore, the total number of requests counted by the hardware are summed up and reported as the Total L3 Cache Bandwidth from the rest of the system (EvtID 2). The number of requests counted by the hardware that were serviced within the local NUMA domain will be reported as the Local L3 Cache Bandwidth from the rest of the system (EvtID 3). The QM_CTR value returned for either of these events may be converted to total bytes read by multiplying by the L3 Cache Conversion Factor, returned in EBX by the CPUID function 0xF, ECX=1. To obtain a bandwidth, software should read the Time Stamp Counter (TSC) and the L3 Cache Bandwidth count, wait some period of time and then read the TSC and the L3 Cache Bandwidth count again. Software should then divide the difference between the count values by the difference in the TSC times returned.

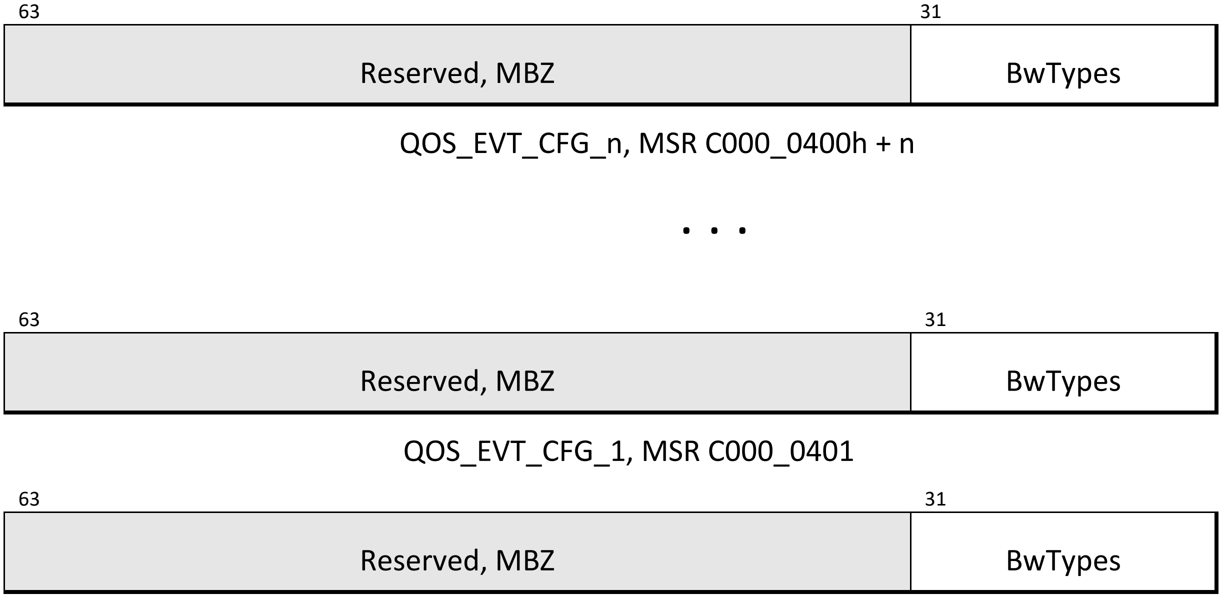
If BMEC is implemented, as determined by CPUID, the types of system requests counted in each event are configurable. CPUID Fn8000_0020_EBX_x4.EVT_NUM identifies the number of configurable events available in the implementation. For example, if the EVT_NUM value is 2, the system supports BMEC Events 0 and 1. The following table associates BMEC number, the register used to configure it, and the EvtID used to access it.

BMEC Event Translation

BMEC Event	Configuration MSR	EvtID	Default Configuration
0	QOS_EVT_CFG_0	2	Defaults to track Total L3 Cache Bandwidth
1	QOS_EVT_CFG_1	3	Defaults to track Reads and Non-Temporal Writes to the Local NUMA Domain

Each BMEC event has a configuration MSR, QOS_EVT_CFG_<EVT> which contains one field for each Bandwidth Type identified in CPUID Fn8000_0020_ECX_x4 that can be used to configure the Bandwidth Event to track any combination of supported bandwidth types. The event will count requests from every Bandwidth Type bit that is set in the corresponding configuration register. Bit locations that are 0 in the CPUID field are Reserved and MBZ in the Configuration Register. Each event will default to track the memory types identified by the corresponding EvtID when BMEC is not supported.

QOS_EVT_CFG_<EVT> Register Definition, MSR C000_0400h + EVT



QOS_EVT_CFG_n, MSR C000_0400h + n

QOS_EVT_CFG_1, MSR C000_0401

QOS_EVT_CFG_0, MSR C000_0400h

Bits	Mnemonic	Description	R/W
63:32	Reserved	Reserved, MBZ	
31:0	BwTypes	BwTypes to track with this event. See CPUID Fn8000_0020_ECX_x4 for individual bit definitions.	R/W

QOS_EVT_CFG_<EVT> MSR Registers

When an Event Configuration is changed, the bandwidth counters for all RMIDs and Events will be cleared, and the U-bit for every RMID will be set on the next read to any BwEvent for every RMID.

A given implementation may have insufficient hardware to simultaneously track the bandwidth for all RMID values which the hardware supports. If an attempt is made to read a Bandwidth Count for an RMID that has been impacted by these hardware limitations, the “U” bit of the QM_CTR will be set when the counter is read. Subsequent QM_CTR reads for that RMID and Event may return a value with the "U" bit clear. Potential causes of the “U” bit being set include (but are not limited to):

- RMID is not currently tracked by the hardware.
- RMID was not tracked by the hardware at some time since it was last read.
- RMID has not been read since it started being tracked by the hardware.

All RMIDs which are currently in use by one or more processors in the QOS domain will be tracked. The hardware will always begin tracking a new RMID value when it gets written to the PQR_ASSOC register of any of the processors in the QOS domain and it is not already being tracked. When the hardware begins tracking an RMID that it was not previously tracking, it will clear the QM_CTR for all events in the new RMID.

In PQoS Version 2.0 or higher, when the hardware begins tracking an RMID that it was not previously tracking, it will set the “U” bit on the first QM_CTR read from that RMID. The “U” bit will be zero for all subsequent reads from that RMID while it is still tracked by the hardware. Therefore, a QM_CTR read with the “U” bit set when that RMID is in use by a processor can be considered 0 when calculating the difference with a subsequent read. The “U” bit will not be set when a counter rolls over. In these implementations, if the QM_CTR read for a given RMID and event shows the "U" bit clear, then the hardware has been counting the event for that RMID continuously without interruption since the previous QM_CTR read for that RMID and that event, as long as the samples were taken in a way to account for a roll-over on the counter.

QOS Enforcement

QOS enforcement is accomplished by assigning a Class Of Service (COS) to a processor and specifying allocations or limits for that COS for each resource to be allocated. The current COS for a given processor is specified in the PQR_ASSOC MSR, described above. If multiple processors within a QOS domain are assigned the same COS, then the resource allocation associated with that class will be shared among all the processors. At reset, the PQR_ASSOC.COS value is 0.

For each resource which is managed by the Platform QOS Enforcement feature, a bank of registers is defined which software can program with the allocation limits for each COS. The interpretation of that register depends on the type of resource which is being managed.

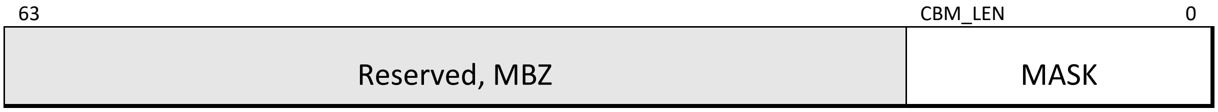
Platform QOS Limits for Cache Allocation Enforcement

The PQE limits for L3 cache allocation are specified by a bank of MSRs called L3_MASK_n, where “n” is the COS. These registers begin with MSR C90h. There is one register for each COS implemented for that resource. Each of the registers is a bitmask with the MSB at CBM_LEN, which is returned in EAX[4:0] by CPUID Fn0000_0010, EAX=1. (This length is zero-based, so the actual number of QOS bits is EAX[4:0] + 1).

Software programs these registers with a bit mask where each “1” represents a portion of the cache which may be used by the corresponding COS. For example, if CBM_LEN for a given implementation is 15, then each “1” which is set in L3_MASK_n represents 1/16 of the cache which may be used by processors running with COS = n. If two or more different Classes of Service have a “1” set in the same bit position in their respective L3_MASK_n register, that represents a portion of the cache which is competitively shared by processors running with those COS values. Some products may implement Cache Allocation Enforcement by allocating some number of ways of the L3 cache for each “1” in the L3_MASK register, but this will not necessarily be true for all implementations and software should not rely on that interpretation. However, because this is one possible implementation, it is possible that use of PQE for cache allocation will reduce the effective associativity available to processes running using an L3_MASK which does not have all the bits set. The bits which are set in the various L3_MASK_n registers do not have to be contiguous and may overlap in any desired combination. If an L3_MASK_n register is programmed with all 0’s, that COS will be prevented from allocating any lines in the L3 cache. At reset, all L3_MASK_n registers are initialized to all 1’s, allowing all processors to use the entire L3 cache accessible to them.

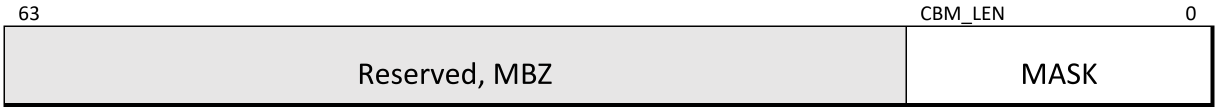
The L3 Cache allocation sharing mask (L3ShareAllocMask) returned in EBX by CPUID Fn0000_0010 with ECX=1 is a bit vector which represents portions of the cache which may be shared with other system entities or used for some other purpose not under the control of the QOS feature set. When software sets a bit in one of the L3_MASK_n registers at the same bit positions a bit in the L3ShareAllocMask, processors executing with the corresponding COS will competitively share that portion of the cache with the other function. If this mask is all 0’s, then there is no other entity in the system competing with the processors for use of the L3 cache.

When a single cache line is accessed by processors running with different COS values in the same COS domain, only a single copy of the line will be allocated in the shared cache at a given time. Depending on the access pattern, this line may be installed in the cache under either of the COS values and will only be counted as part of that COS allocation. Depending on subsequent access patterns, that cache line may later be counted against the other COS allocation, but software should not assume that the line will always be associated with the last COS which accessed the line.

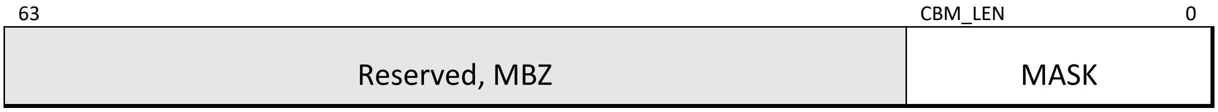


L3_MASK_n, MSR C90h + n

• • •



L3_MASK_1, MSR C91h



L3_MASK_0, MSR C90h

Bits	Mnemonic	Description	R/W
63:CBM_LEN+1	Reserved	Reserved, MBZ	
CBM_LEN:0	MASK	L3 Cache PQE Allocation Mask	R/W

L3_MASK_<COS> MSR Registers

Code and Data Prioritization (CDP)

If CPUID function 0x10, ECX = 1 indicates that CDP is supported, software may enable the CDP mode of PQE by setting bit 0 of L3_QOS_CFG1, MSR C81h. In this mode, software may specify a different cache allocation mask for instructions versus data. A given process is still only assigned to a single COS, but a given COS will use two different mask registers. The data allocation mask will be specified in MSR C90 + (2*COS) while the instruction allocation mask will be specified by MSR C91 +(2*COS). Note that enabling CDP cuts in half the number of unique COS values which may be specified.

The interpretation of the L3_MASK registers is the same with CDP on as with it off; the difference is that the mask to be applied is selected based on whether the line is accessed as an instruction or data fetch. Similar to the behavior with CDP disabled, if a line is accessed as both code and data, then the line may be allocated using either mask register and be counted against that allocation limit. Also similar to the case with CDP disabled, software may not assume that cache lines will necessarily be associated with the code or data mask register with which they were most recently accessed.

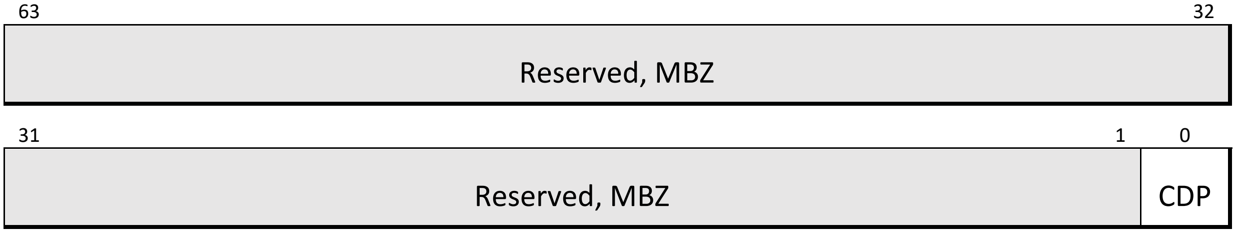
Software should observe the following sequence when enabling or disabling CDP mode:

1. Ensure that the QOS Allocation mask for each COS (L3_MASK_<COS>.MASK) is set to all 1's.
2. Ensure that the QOS Bandwidth enforcement limit for each COS is set to "No Limit". (L3QOS_BW_CONTROL_<COS>.U == 1)
3. Ensure that all logical processors in the QOS domain are associated with Valid COS numbers for when CDP is enabled (PQR_ASSOC.COS). Specifically, the COS number must be less than half of COS_MAX.
4. Write the desired configuration of CDP to the L3_QOS_CFG1 MSR on each logical processor in the QOS domain.

For optimal QOS behavior in the new operating mode, software should flush the caches in the QOS domain once the new configuration has been enabled to clear out any residual effects from the previous configuration. In some implementations, a single L3_QOS_CFG1 MSR may be shared between multiple logical processors.

	CDP = 0		CDP=1	
COS	Data	Code	Data	Code
0	C90	C90	C90	C91
1	C91	C91	C92	C93
2	C92	C92	C94	C95
3	C93	C93	C96	C97
4	C94	C94	C98	C99
N	C90+N	C90+N	C90+2*N	C91+2*N

L3CACHE_ALLOC_MSR association, CDP mode enabled



Bits	Mnemonic	Description	R/W
63:1	Reserved	Reserved, MBZ	
0	CDP	Code and Data Prioritization Enable	R/W

L3_QOS_CFG1, MSR C81h

Platform QOS Limits for Bandwidth Enforcement

Similar to PQE for cache allocation, bandwidth limits are specified in a bank of MSRs called L3QOS_BW_CONTROL_n, where "n" is the corresponding COS number. These MSRs begin at C000_0200h.

The value programmed in the L3QOS_BW_CONTROL_n register specifies a limit on the total bandwidth counted by the QOS hardware, which the collection of all processors running with that COS and within that COS domain may consume. Unlike cache allocation (which specifies a fraction of the cache), this limit is not a relative bandwidth, but an absolute number expressed in 1/8 GB/s increments. The maximum value which can be programmed in this register may exceed the total bandwidth which the system can actually supply to that processor. Software may specify "no limit" by programming a "1" into the "U" field of the L3QOS_BW_CONTROL_n register which is located in the BW_LEN+1 bit position. That is, if BW_LEN = 0xb, the maximum limit which could be programmed would be 0x7FF and "unlimited" may be specified by 0x800. If the "U" bit is set, the other bits in the BW field have no effect. At reset, the "U" bit of all L3QOS_BW_CONTROL_n registers is set to 1, allowing maximum bandwidth usage by all processors. Bandwidth limits are upper bounds on the bandwidth the processors in a given domain and a given COS may consume, but do not ensure that the specified bandwidth will necessarily be available to the processors running with a given COS. Cases where the processors in a given class may not be able to reach their allocated bandwidth include (but are not limited to):

1. Maximum system bandwidth may be less than the specified limit.
2. The sum of the limits applied to all classes of service in the domain may exceed the maximum bandwidth the system can deliver to that COS domain.
3. Multiple COS domains which share the same memory channels may demand more total bandwidth than the shared memory can supply.

4. I/O or other system entities may consume a large fraction of system bandwidth and result in less bandwidth being available to the various CPU COS domains.
5. Large amounts of write traffic may affect the memory system’s ability to deliver read bandwidth.

63		BW_LEN-1 0
Reserved, MBZ	U	BW

L3QOS_BW_CONTROL_n, MSR C000_0200h + n

• • •

63		BW_LEN-1 0
Reserved, MBZ	U	BW

L3QOS_BW_CONTROL_1, MSR C000_0201

63		BW_LEN-1 0
Reserved, MBZ	U	BW

L3QOS_BW_CONTROL_0, MSR C000_0200h

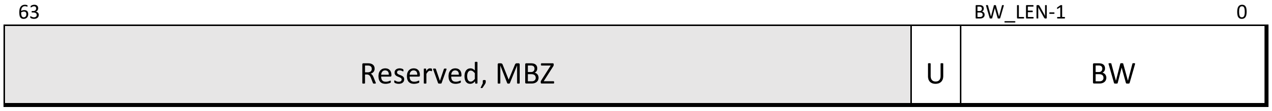
Bits	Mnemonic	Description	R/W
63:BW_LEN+1	Reserved	Reserved, MBZ	
BW_LEN	U	Unlimited bandwidth	R/W
BW_LEN-1:0	BW	L3 Cache PQE External Bandwidth Limit (1/8 GB/s)	R/W

L3QOS_BW_CONTROL_<COS> MSR Registers

In the cases above where not all COSs can reach their bandwidth limits, the hardware makes no attempt to allocate the available memory bandwidth in proportion to the limits specified for each COS domain or the limits specified for each COS within a given domain. That is (in case2), if COS x has limit A, and COS y has limit 2*A, and the total bandwidth which the system can deliver to that COS is only 2*A, the system will only limit COS x to A; it will not attempt to give (2/3)*A to COS x and (4/3)*A to COS y (maintaining the ratio of their limits).

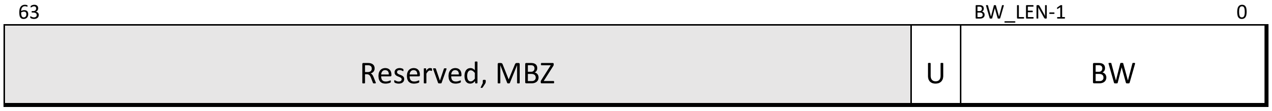
If L3 External Slow Memory Bandwidth Allocation Enforcement (L3SBE) is supported in this product (as identified by CPUID), there is an additional bank of MSRs called L3QOS_SLOWBW_CONTROL_n where "n" is the corresponding COS number. These MSRs begin at C000_0280h.

The value programmed in the L3QOS_SLOWBW_CONTROL_n register specifies a limit on the bandwidth to Slow Memory counted by the QOS hardware, which the collection of all processors running with that COS and within that COS domain may consume. The format of these MSR's are identical to the L3QOS_BW_CONTROL_n register except that the size of the BW field is identified by CPUID Fn8000_0020_EAX_x2.BW_LEN.

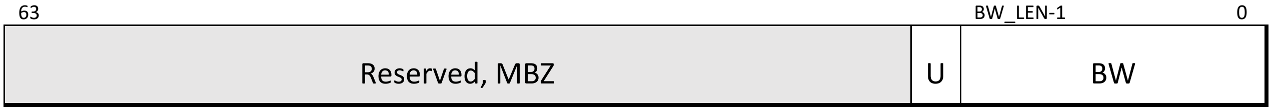


L3QOS_SLOWBW_CONTROL_n, MSR C000_0280h + n

• • •



L3QOS_SLOWBW_CONTROL_1, MSR C000_0281h



L3QOS_SLOWBW_CONTROL_0, MSR C000_0280h

Bits	Mnemonic	Description	R/W
63:BW_LEN+1	Reserved	Reserved, MBZ	
BW_LEN	U	Unlimited bandwidth	R/W
BW_LEN-1:0	BW	L3 Cache PQE Slow Bandwidth Limit (1/8 GB/s)	R/W

L3QOS_SLOWBW_CONTROL_<COS> MSR Registers

The hardware may limit Slow Memory Bandwidth by limiting all memory bandwidth from a given COS. Therefore, the Slow Memory Bandwidth limit may impact the Total Memory Bandwidth observed by a given COS.