Ncore Perf Counter Architecture Specification

Release: 1.0

Rev: 0.81, February 9, 2024

Release Information

Version	*Editor*	*Change*	*Date*
0.5	MK	Initial release from 3.2 with updates Added ACE-Lite Snoop stall events Added events for BW counters Added universal single trigger in DVE	02/04/2022
0.6	MK	Updates to add in Latency counters	02/07/2022
0.7	MK	Updated trigger specification and latency register	02/02/2022
0.75	MK	Added BW filters based on feedback from ME Changed the number of counters parameter to per unit and expanded the range	03/10/2022
0.77	MK	Updated BW and Latency counters based on feedback Added separate latency counter register Updated count control registers and BW/Latency count procedures Fixed typos	03/30/2022
0.8	MK	Updated Trigger register description	09/02/2022
0.81	JV	Updated Performance Event Tables to clarify event number for cache events on 1.4.3 & 1.4.5. Added detail on xCNTCR[Count event first] & xCNTCR[Count event second] fields.	02/09/2024
Legend:	MK MF JV Xx	Mohammed Khaleeluddin Michael Frank Jason Villanueva Whoever else edited this document	

Confidential Proprietary Notice

extension of existing patents or patent applications; or (iv) for generating data for publication or disclosure to third parties, which compares the performance or functionality of the Arteris IP technology described in this document with any other products created by you or a third party, without obtaining Arteris IP's prior written consent.

THIS DOCUMENT IS PROVIDED "AS IS". ARTERIS IP PROVIDES NO REPRESENTATIONS AND NO WARRANTIES, EXPRESS, IMPLIED OR STATUTORY, INCLUDING, WITHOUT LIMITATION, THE IMPLIED WARRANTIES OF MERCHANTABILITY, SATISFACTORY QUALITY, NON-INFRINGEMENT OR FITNESS FOR A PARTICULAR PURPOSE WITH RESPECT TO THE DOCUMENT. For the avoidance of doubt, Arteris IP makes no representation with respect to, and has undertaken no analysis to identify or understand the scope and content of, third party patents, copyrights, trade secrets, or other rights. This document may include technical inaccuracies or typographical errors. Arteris IP makes no representations or warranties against the risk or presence of same.

TO THE EXTENT NOT PROHIBITED BY LAW, IN NO EVENT WILL ARTERIS IP BE LIABLE FOR ANY DAMAGES, INCLUDING WITHOUT LIMITATION ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL, PUNITIVE, OR CONSEQUENTIAL DAMAGES, HOWEVER CAUSED AND REGARDLESS OF THE THEORY OF LIABILITY, ARISING OUT OF ANY USE OF THIS DOCUMENT, EVEN IF ARTERIS HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

This document consists solely of commercial items. You shall be solely responsible for ensuring that any use, duplication or disclosure of this document complies fully with any relevant export laws and regulations to assure that this document or any portion thereof is not exported, directly or indirectly, in violation of such export laws. Use of the word "partner" in reference to Arteris IP's customers is not intended to create or refer to any partnership relationship with any other company. Arteris IP may make changes to this document at any time and without notice. If any of the provisions contained in these terms conflict with any of the provisions of any click through or signed written agreement covering this document with Arteris IP, then the click-through or signed written agreement prevails over and supersedes the conflicting provisions of these terms. This document may be translated into other languages for convenience, and you agree that if there is any conflict between the English version of this document and any translation, the terms of the English version of the agreement shall prevail.

Confidentiality Status

This document is Confidential and Proprietary. This document may only be used and distributed in accordance with the terms of the agreement entered into by Arteris IP and the party that Arteris IP delivered this document to.

Product Status

The information in this document is ***Preliminary***.

Web Address

http://www.arteris.com

Table of Contents

Table of Figures

Table of Tables

Preface

This preface introduces the Arteris® Network-on-Chip Hierarchical Coherency Engine Architecture Specification.

About this document

This technical document is for the Arteris Network-on-Chip Hierarchical Coherency Engine Architecture. It describes the subsystems and their function along with the system's interactions with the external subsystems. It also provides reference documentation and contains programming details for registers.

Product revision status

TBD

Intended audience

This manual is for system designers, system integrators, and programmers who are designing or programming a System-on-Chip (SoC) that uses or intend to use the Arteris Network-on-Chip Hierarchical Coherency System (ANoC-HCS).

Using this document

TBD

Glossary

The Arteris© Glossary is a list of terms used in Arteris© documentation, together with definitions for those terms. The Arteris© Glossary does not contain terms that are industry standard unless the Arteris© meaning differs from the generally accepted meaning.

Typographic conventions

italic

Introduces special terminology, denotes cross-references, and citations.

bold

Highlights interface elements, such as menu names. Denotes signal names. Also used for terms in descriptive lists, where appropriate.

monospace

Denotes text that you can enter at the keyboard, such as commands, file and program names, and source code.

monospace italic

Denotes a permitted abbreviation for a command or option. You can enter the underlined text instead of the full command or option name. monospace italic Denotes arguments to monospace text where the argument is to be replaced by a specific value. monospace bold Denotes language keywords when used outside example code.

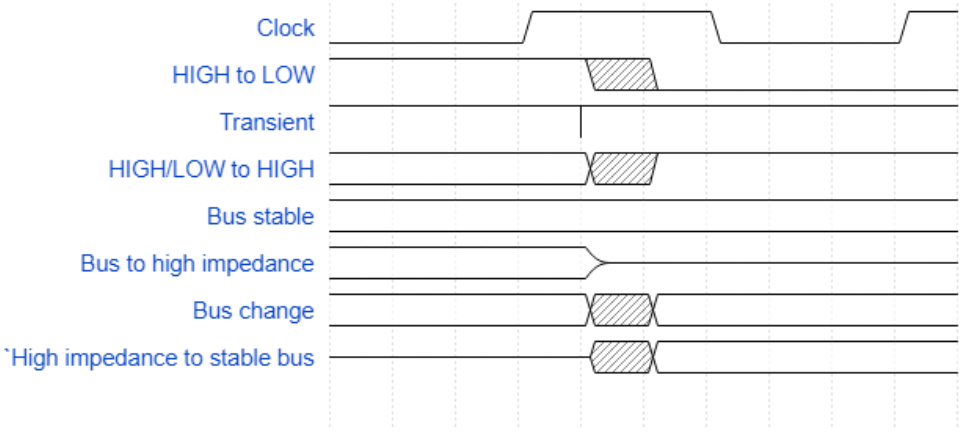
SMALL CAPITALS

Used in body text for a few terms that have specific technical meanings, that are defined in the Arteris® Glossary. For example, IMPLEMENTATION DEFINED, IMPLEMENTATION SPECIFIC, UNKNOWN, and UNPREDICTABLE.

Timing diagrams

The following figure explains the components used in timing diagrams. Variations, when they occur, have clear labels. You must not assume any timing information that is not explicit in the diagrams.

Shaded bus and signal areas are undefined, so the bus or signal can assume any value within the shaded area at that time. The actual level is unimportant and does not affect normal operation.



Signals

The signal conventions are:

Signal level

The level of an asserted signal depends on whether the signal is active-HIGH or active-LOW. Asserted means:

- HIGH for active-HIGH signals.
- LOW for active-LOW signals.

Lowercase n

At the start or end of a signal name denotes an active-LOW signal.

Additional reading

This book contains information that is specific to this product. See the following documents for other relevant information.

History of the World II, Mel Brooks.

1 Introduction

This specification describes the performance counter architecture for Ncore.

A performance counter unit must be implemented in each Ncore unit i.e. AIUs, DCEs, DMIs, DIIs and DVE. This document goes over the architecture details and parameters of this unit and different events that are reported. The performance counter unit is not an optional unit and must always be present.

1.1 Parameters

Two new unit level parameters are introduced, they are shown in Table 1 & Table 2.

Name: nPerfCounters **Type:** Int **Visibility:** user Settable

	Architecture		Release		Default
	Min	*Max*	*Min*	*Max*	
Value	0	16	0	16	4
Constraint	Only valid values are 0, 4, 8 and 16				
Customer Description	Number of performance counters per Ncore unit				
Engineering Description					

TABLE 1 NPERFCOUNTERS PARAMETER

Name: nLatencyCounters **Type:** Int **Visibility:** user Settable

	Architecture		Release		Default
	Min	*Max*	*Min*	*Max*	
Value	0	32	0	16	16
Constraint	Only two valid values are supported 0 or 16. A non-zero value is possible only if nPerfCounters is greater than or equal to 4				
Customer Description	Number of Latency counters in the Ncore unit.				
Engineering Description	Parameter applies only to AIUs, DMIs and DIIs only and can be set individually				

TABLE 2 NLATENCYCOUNTERS PARAMETER

1.2 Detailed Description

The performance counter unit can track and report upto 31 events. The Ncore unit instantiating the performance unit specifies the events. These events maybe single bit or multi bit events and are specified in 1.4 section. The performance counter unit can be configured to implement either 4 or 8 counters.

Each counter is implemented as a 64 bit counter that is capable of counting upto 2 events at a time with following counter modes.

1. **Normal count**: In this mode the counter counts both the events together, if both the events are asserted then it is counted as two, if one event is asserted then it is counted as one
2. **AND count**: In this mode the counter counts one if both the events are asserted.
3. **XOR count**: In this mode the counter counts one only if one event is asserted and the other is de asserted.
4. **Instantaneous count:** In this mode the counter provides the instantaneous value of the multi bit event as is.
5. **32-bit count:** In this mode the 64 bit counter acts as two 32 bit counters for two separate events

The counter reports count via 2 registers one is a 32-bit count register that reports lower 32 bits and another configurable register which supports

1. Capturing the upper 32 bits of the count
2. Used as an accumulation register for IIR filter
3. Use as max/saturation value for accumulation events
4. Use as 32 bit counter for second event

Both count registers freeze when an overflow is detected at either of the two registers, with the exception when they are configured to work in tandem as a 64 bit counter.

1.3 Register Definition

Different registers supported by the performance counter unit are described below. A set of registers exist per counter configuration i.e. each Ncore unit can either have 4 or 8 set of registers. Note that xMCNTCR and xLCNTCR is only one per Ncore unit.

1.3.1 Counter Value Register (xCNTVR)

Bits	Name	Access	Reset	Description
31:0	Count value	RO	0x0	Count value lower bits 31:0, freeze at overflow.

1.3.2 Counter Saturation Register (xCNTSR)

Bits	Name	Access	Reset	Description
31:0	Count saturation value	RO	0x0	Can be configured as follows (freeze at overflow): 1. Capture upper count value of 63:31 2. Use as IIR filter bits 3. Use as max/saturation value for accumulation events 4. Use as 32-bit counter, for count event second

1.3.3 Counter Control Register (xCNTCR)

Bits	Name	Access	Reset	Description
0	Count Enable	R/W	0x0	Write one to enable counting, this may be set by the “Master Count Enable” signal from DVE or can be set or reset by “Local Count Enable”
1	Count Clear	WSC	0x0	Write one to clear the counter (both CNTVR and CNTSR)
2	Interrupt Enable	R/W	0x0	Write one to enable rollover or overflow interrupt (gets ORed with correctable error interrupt)
3	Rollover/Overflow status	RO	0x0	One indicates overflow, sticky bit clears by clearing the counter. (When either xCNTVR or xCNTSR overflow, with exceptions as specified above)
6:4	Counter control	R/W	0x0	000 – Normal count 001 – AND count 010 – XOR count 011 – Instantaneous count (used for multi bit events like Active OTT entries) 100 – use as 32-bit counter (count event first) 100 to 111 – reserved
9:7	SSR count	R/W	0x0	000 – Clear CNTSR (value of CNTSR is always zero) 001 – Capture upper 63:31 count in CNTSR 010 – use CNTSR as IIR filter (currently used only for active TT count) 011 – use CNTSR as max/saturation value (currently used only for CHI AIU interleave count) 100 – use as 32-bit counter (count event second) 101 to 111 reserved
12:10	Filter select	R/W	0x0	Low pass filter coefficients (IIR filter) 000 – 0 001 – ½ 010 – ¼ 011 – 1/8 100 – 1/16 101 – 1/32 110 – 1/64 111 – 1/128
15:13	Minimum stall period	R/W	0x0	Value is 2 ^ (minimum stall period) clock cycles valid range is 0 to 7
16:21	Count event second	R/W	0x0	Select the second count event number (must be different from Count event first), value of zero is not valid, if zero no second event is selected. This value is the event # provided in the events table.
22:23	Reserved			
24:29	Count event first	R/W	0x0	Select the first count event number, value of zero is not valid, if zero no event is selected. This value is the event # provided in the events table.
31:30	Reserved			

1.3.4 BW Counter Filter Register (xBCNTFR)

Bits	Name	Access	Reset	Description
19:0	Filter value	R/W	0x0	Value of user bits or Funit ID (as selected by filter select) to be filtered on. Must be LSB aligned
29:20	Reserved			
30	Filter select	R/W	0x0	Set to filter BW counting on Funit ID, else filter on user bits. This is valid only if Filter enable is set
31	Filter enable	R/W	0x0	Set to enable BW counting filtering else disabled

1.3.5 BW Counter Mask Register (xBCNTMR)

Bits	Name	Access	Reset	Description
19:0	Mask value	R/W	0x0	Mask for user bits or Funit ID (as selected by filter select) to be filtered on. Must be LSB aligned
31:20	Reserved			

1.3.6 Main Counter Control Register (xMCNTCR)

Bits	Name	Access	Reset	Description
0	Local Count Enable	R/W	0x0	Enables all local counters when set (0 -1) and stops all local counters when reset (1 to 0)
1	Local Count Clear	WSC	0x0	Clear all local counters
30:2	Reserved			
31	Master Count Enable	R/W	0x0	Only present in DVE and is reserved for all other blocks. Setting this (0 to 1) will set Count Enable in all Ncore unit performance counters and resetting (1 to 0) will reset Count Eanble in all Ncore unit performance counters i.e., stop counting in all Ncore units

1.3.7 Latency Counter Control Register (xLCNTCR)

Present if latency counters parameter is set

Bits	Name	Access	Reset	Description
1:0	Latency pre scale	R/W	0x0	0b00 – count every 2 cycles 0b01 – count every 4 cycles 0b10 – count every 8 cycles 0b11 – count every 16 cycles
3:2	Reserved			
4	Latency Count Enable	R/W	0x0	Enable latency counter
5	Read/Write Latency	R/W	0x0	Set ‘0’ to count read latency Set ‘1’ to count write latency
7:6	Reserved			
15:8	Latency bin offset	R/W	0x0	8-bit offset value to be subtracted from latency counter
30:16	Reserved			

1.4 Performance events

1.4.1 CAIU Performance events

Event #	Width	Name	Description
1	1	SMI 0 Tx Stall event	Counts every cycle when valid is set and ready is low
2	1	SMI 1 Tx Stall event	Counts every cycle when valid is set and ready is low
3	1	SMI 2 Tx Stall event	Counts every cycle when valid is set and ready is low
4		Reserved	
5	1	SMI 0 Rx Stall event	Counts every cycle when valid is set and ready is low
6	1	SMI 1 Rx Stall event	Counts every cycle when valid is set and ready is low
7	1	SMI 2 Rx Stall event	Counts every cycle when valid is set and ready is low
8		Reserved	
9	1	ACE AW stall event	Counts every cycle when valid is set and ready is low (Does not apply to CHI AIU)
10	1	ACE W stall event	Counts every cycle when valid is set and ready is low (Does not apply to CHI AIU)
11	1	ACE B stall event	Counts every cycle when valid is set and ready is low (Does not apply to CHI AIU)
12	1	ACE AR stall event	Counts every cycle when valid is set and ready is low (Does not apply to CHI AIU)
13	1	ACE R stall event	Counts every cycle when valid is set and ready is low (Does not apply to CHI AIU)
14	1	ACE AC stall event	Counts every cycle when valid is set and ready is low (Does not apply to CHI AIU)
15	1	ACE CD stall event	Counts every cycle when valid is set and ready is low (Does not apply to CHI AIU)
16	1	ACE CR stall event	Counts every cycle when valid is set and ready is low (Does not apply to CHI AIU)
17	1	CmdReq WR event	Count all write commands that are associated with data, refer CCMP for all related commands.
18	1	CmdReq RD event	Count all read commands that are associated with data, refer CCMP for all related commands.
19	1	SnpRsp event	Count all snoop responses that trigger data transfer read or write.
20	8	Active OTT entries	Number of active OTT entries
21		Reserved	
22	3	Captured SMI packets	Number of SMI packets Captured
23	3	Dropped SMI packets	Number of SMI packets dropped
24	1	Address Collisions	Count number of address collisions on incoming transactions
25	3	Interleaved Data	Count max number of active interleaved data transactions
26	1	Agent event counter	Counts number of events triggered by the native agent

27	1	Noc evet counter	Counts number of events triggered by the Noc
28		Reserved	
29		Reserved	
30	1	Div 16 counter	Divide by 16 free running counter
31	1	Number of QoS starvations	Number of times QoS starvations occurred

1.4.2 NCAIU Performance events

Event #	Width	Name	Description
1	1	SMI 0 Tx Stall event	Counts every cycle when valid is set and ready is low
2	1	SMI 1 Tx Stall event	Counts every cycle when valid is set and ready is low
3	1	SMI 2 Tx Stall event	Counts every cycle when valid is set and ready is low
4		Reserved	
5	1	SMI 0 Rx Stall event	Counts every cycle when valid is set and ready is low
6	1	SMI 1 Rx Stall event	Counts every cycle when valid is set and ready is low
7	1	SMI 2 Rx Stall event	Counts every cycle when valid is set and ready is low
8		Reserved	
9	1	ACE-Lite/AXI AW stall event	Counts every cycle when valid is set and ready is low
10	1	ACE-Lite/AXI W stall event	Counts every cycle when valid is set and ready is low
11	1	ACE-Lite/AXI B stall event	Counts every cycle when valid is set and ready is low
12	1	ACE-Lite/AXI AR stall event	Counts every cycle when valid is set and ready is low
13	1	ACE-Lite/AXI R stall event	Counts every cycle when valid is set and ready is low
14	1	ACE-Lite AC stall event	Counts every cycle when valid is set and ready is low
15		Reserved	
16	1	ACE-Lite CR stall event	Counts every cycle when valid is set and ready is low
17	1	CmdReq WR event	Count all write commands that are associated with data, refer CCMP for all related commands.
18	1	CmdReq RD event	Count all read commands that are associated with data, refer CCMP for all related commands.
19	1	SnpRsp event	Count all snoop responses that trigger data transfer read or write
20	8	Active OTT entries	Number of active OTT entries
21		Reserved	
22	3	Captured SMI packets	Number of SMI packets Captured
23	3	Dropped SMI packets	Number of SMI packets dropped
24	1	Address Collisions	Count number of address collisions on incoming transactions
25		Reserved	
26	1	Agent event counter	Counts number of events triggered by the native agent
27	1	Noc evet counter	Counts number of events triggered by the Noc
28		Reserved	
29		Reserved	
30	1	Div 16 counter	Divide by 16 free running counter

31	1	Number of QoS starvations	Number of times QoS starvations occurred

1.4.3 Proxy Performance events

Event #	Width	Name	Description
32	1	Cache read hit	
33	1	Cache write hit	
34	1	Cache snoop hit	
35	1	Cache eviction	
36	1	Cache no ways to allocate	
37	1	Cache fill stall	
38	1	Cache read stall	
39	1	Cache write stall	
40	1	Cache replay	
41	1	Cache read miss	
42	1	Cache write miss	
43	1	Cache snoop miss	
44		Reserved	
45		Reserved	
46		Reserved	
47		Reserved	
48		Reserved	
49		Reserved	
50		Reserved	
51		Reserved	
52		Reserved	
53		Reserved	
54		Reserved	
55		Reserved	
56		Reserved	
57		Reserved	
58		Reserved	
59		Reserved	
60		Reserved	
61		Reserved	
62		Reserved	

1.4.4 DMI Performance events

Event #	Width	Name	Description
1	1	SMI 0 Tx Stall event	Counts every cycle when valid is set and ready is low
2	1	SMI 1 Tx Stall event	Counts every cycle when valid is set and ready is low

3	1	SMI 2 Tx Stall event	Counts every cycle when valid is set and ready is low
4	1	SMI 3 Tx Stall event	Counts every cycle when valid is set and ready is low
5	1	SMI 0 Rx Stall event	Counts every cycle when valid is set and ready is low
6	1	SMI 1 Rx Stall event	Counts every cycle when valid is set and ready is low
7	1	SMI 2 Rx Stall event	Counts every cycle when valid is set and ready is low
8	1	SMI 3 Rx Stall event	Counts every cycle when valid is set and ready is low
9	1	AXI AW stall event	Counts every cycle when valid is set and ready is low
10	1	AXI W stall event	Counts every cycle when valid is set and ready is low
11	1	AXI B stall event	Counts every cycle when valid is set and ready is low
12	1	AXI AR stall event	Counts every cycle when valid is set and ready is low
13	1	AXI R stall event	Counts every cycle when valid is set and ready is low
14		Reserved	
15		Reserved	
16		Reserved	
17	1	DtwReq event	Count all DtwReqs that come into DMI
18	1	DtrReq event	Count all DtrReqs that are issued by DMI
19		Reserved	
20	8	Active WTT entries	Number of active WTT entries
21	8	Active RTT entries	Number of active RTT entries
22	4	Captured SMI packets	Number of SMI packets Captured
23	4	Dropped SMI packets	Number of SMI packets dropped
24	1	Address Collisions	Count number of address collisions on incoming transactions
25	1	Number of Merge events	Count number of DtwMergeMrds
26	1	Number of system visible Txn	Count number of system visible transactions
27		Reserved	
28		Reserved	
29		Reserved	
30	1	Div 16 counter	Divide by 16 free running counter
31	1	Number of QoS starvations	Number of times QoS starvations occurred

1.4.5 SMC Performance events

Event #	Width	Name	Description
32	1	Cache read hit	
33	1	Cache write hit	
34	1	Cache CMO hit	Operations like cleanInvlidate etc
35	1	Cache eviction	
36	1	Cache no ways to allocate	
37	1	Cache fill stall	
38	1	Cache read stall	
39	1	Cache write stall	
40	1	Cache replay	
41	1	Cache read miss	

42	1	Cache write miss	
43	1	Cache CMO miss	
44		Reserved	
45		Reserved	
46		Reserved	
47		Reserved	
48		Reserved	
49		Reserved	
50		Reserved	
51		Reserved	
52		Reserved	
53		Reserved	
54		Reserved	
55		Reserved	
56		Reserved	
57		Reserved	
58		Reserved	
59		Reserved	
60		Reserved	
61		Reserved	
62		Reserved	

1.4.6 DII Performance events

Event #	Width	Name	Description
1	1	SMI 0 Tx Stall event	Counts every cycle when valid is set and ready is low
2	1	SMI 1 Tx Stall event	Counts every cycle when valid is set and ready is low
3	1	SMI 2 Tx Stall event	Counts every cycle when valid is set and ready is low
4		Reserved	
5	1	SMI 0 Rx Stall event	Counts every cycle when valid is set and ready is low
6	1	SMI 1 Rx Stall event	Counts every cycle when valid is set and ready is low
7	1	SMI 2 Rx Stall event	Counts every cycle when valid is set and ready is low
8		Reserved	
9	1	AXI AW stall event	Counts every cycle when valid is set and ready is low
10	1	AXI W stall event	Counts every cycle when valid is set and ready is low
11	1	AXI B stall event	Counts every cycle when valid is set and ready is low
12	1	AXI AR stall event	Counts every cycle when valid is set and ready is low
13	1	AXI R stall event	Counts every cycle when valid is set and ready is low
14		Reserved	
15		Reserved	
16		Reserved	
17	1	DtwReq event	Count all DtwReqs that come into DII
18	1	DtrReq event	Count all DtrReqs that are issued by DII
19		Reserved	

20	8	Active WTT entries	Number of active WTT entries
21	8	Active RTT entries	Number of active RTT entries
22	3	Captured SMI packets	Number of SMI packets Captured
23	3	Dropped SMI packets	Number of SMI packets dropped
24	1	Address Collisions	Count number of address collisions on incoming transactions
25		Reserved	
26		Reserved	
27		Reserved	
28		Reserved	
29		Reserved	
30	1	Div 16 counter	Divide by 16 free running counter
31		Reserved	

1.4.7 DCE Performance events

Event #	Width	Name	Description
1	1	SMI 0 Tx Stall event	Counts every cycle when valid is set and ready is low
2	1	SMI 1 Tx Stall event	Counts every cycle when valid is set and ready is low
3	1	SMI 2 Tx Stall event	Counts every cycle when valid is set and ready is low
4		Reserved	
5	1	SMI 0 Rx Stall event	Counts every cycle when valid is set and ready is low
4	1	SMI 1 Rx Stall event	Counts every cycle when valid is set and ready is low
7	1	SMI 2 Rx Stall event	Counts every cycle when valid is set and ready is low
8		Reserved	
9		Reserved	
10		Reserved	
11		Reserved	
12		Reserved	
13		Reserved	
14		Reserved	
15		Reserved	
16		Reserved	
17		Reserved	
18		Reserved	
19		Reserved	
20	8	Active ATT entries	Number of active ATT entries
21		Reserved	
22		Reserved	
23		Reserved	
24	1	Address Collisions	Count number of address collisions on incoming transactions
25	1	SF hit	Snoop filter hit (either owner or sharer) count
26	1	SF miss	Snoop filter miss (neither owner nor sharer) count

27	1	SF recall	Snoop filter recall transaction count
28	1	Snoop rsp miss	Snoop response reports miss
29	1	Snoop rsp Owner transfer	Snoop response Ownership transfer
30	1	Div 16 counter	Divide by 16 free running counter
31	1	Number of QoS Starvations	Number of times QoS starvations occurred

1.4.8 DVE Performance events

Event #	Width	Name	Description
1	1	SMI 0 Tx Stall event	Counts every cycle when valid is set and ready is low
2	1	SMI 1 Tx Stall event	Counts every cycle when valid is set and ready is low
3	1	SMI 2 Tx Stall event	Counts every cycle when valid is set and ready is low
4		Reserved	
5	1	SMI 0 Rx Stall event	Counts every cycle when valid is set and ready is low
4	1	SMI 1 Rx Stall event	Counts every cycle when valid is set and ready is low
7	1	SMI 2 Rx Stall event	Counts every cycle when valid is set and ready is low
8		Reserved	
9		Reserved	
10		Reserved	
11		Reserved	
12		Reserved	
13		Reserved	
14		Reserved	
15		Reserved	
16		Reserved	
17		Reserved	
18		Reserved	
19		Reserved	
20	8	Active STT entries	Number of active STT entries
21		Reserved	
22	1	Captured DtwDbgReq packets	Number of DtwDbg packets captured
23	1	Dropped DtwDbgReq packets	Number of DtwDbg packets dropped
24		Reserved	
25		Reserved	
26		Reserved	
27		Reserved	
28		Reserved	
29		Reserved	
30	1	Div 16 counter	Divide by 16 free running counter
31		Reserved	

1.5 Master Trigger

This acts as a single trigger to start/stop all performance counters in Ncore at approximately the same time. This is achieved by adding a Master count enable bit in DVE. When the bit is set it must be propagated to all Ncore units via a single wire. This signal is expected to be active high asynchronous signal, which is synchronized by the receiving Ncore unit. The Ncore unit will detect rising edge to start the counters and falling edge to stop the counters.

It is expected that SW will set this control register bit then read it to verify that it is set, wait from some x amount of time, and then reset it.

1.6 Bandwidth Counters

Ncore provides Bandwidth counters in AIUs, DMIs and DIIs. These counters count in 64Byte data accuracy and at divided by 16 accuracies of the clock frequency. BW counting can be filtered based on FunitID or user bits, this can be configured in CSRs xBCNTFR and xBCNTMR

In AIUs they count following

- Read data bandwidth, this refers to data being read into the master/cache
- Write data bandwidth, this refers to data being written out of the master/cache, this may include evicted data.
- Snoop data bandwidth, this refers to data being moved out of the master/cache due to snoops

In DMIs/DIIs the count following

- Read data bandwidth, this refers to data being read from DMI/DII. In the case of DMI the data maybe read from SMC if present.
- Write data bandwidth, this refers to data being written into DMI/DII. In the case of DMI the data maybe written into SMC if present.

BW counting is achieved by setting xCTCR0 as follows:

- ***Counter control*** to 3b100 i.e., use as 32 bit counter
- ***SSR count*** to 3b100 i.e., use as 32 bit counter
- ***Count first event*** to one of the following events depending on what is to be counted
  - CmdReq WR event: to count AIU write BW
  - CmdReq RD event: to count AIU read BW
  - SnpRsp event: to count Snoop BW
  - DtwReq event: to count DMI/DII write BW
  - DtrReq event: to count DMI/DII read BW
- ***Count Second event*** to divide by 16 clock cycle event

Once the counter is enabled following is reported

- xCNTSR: number of divide by 16 clock cycles
- xCNTVR: number of 64Bytes of data.

The above information can be used to calculate the effective BW. The counter must be disabled before reading them to get correct BW.

Note that this will be approximate BW as all transactions that were counted may necessarily not be 64 bytes transactions and the time accuracy is divided by 16.

1.7 Latency Histogram Counters

Ncore provides latency counters in AIUs, DMIs and DIIs for either reads or writes. Latency is reported back as a binned histogram. It is an optional feature that is controlled via a unit level parameter as described in Parameters section. At any given time, it can be configured to report only one type of latency i.e., read latency or write latency. The latency reported at AIUs is from Ncore system preceptive i.e., it is reporting the complete lifetime of the transaction within Ncore. The latency reported at DMI/DII is more of a closer representation of latency seen at the Native AXI interface.

Read latency includes all transitions defined by CCMP/Native interface as a read transition that provides data to the requester. Write latency includes all transactions defined by CCMP/Native interface as a write transaction that provides data to be written at the slave. Stash transactions are not included.

Logic for generating the latency histogram is implemented as separate block as shown in Figure 1. In AIUs when the selected type of transaction is allocated to the transaction table (OTT) it is also allocated to the latency counter table, whereas in the case of DMI/DII it is allocated when the transaction is ready to be issued on the Native interface. The ID represents the transaction table (OTT/WTT/RTT) ID and counter is a 9-bit counter. The counter is started as soon as the entry is allocated. If all entries within the latency counter table are taken, then any new qualified transaction just does not participate in the latency histogram.

The counter increments every 2, 4, 8 or 16 clock cycle based on CSR configuration. If the counter hits its max value, then it stays saturated until deallocation and the saturated value is used for latency histogram binning.

In the case of an AIU an entry in latency counter table is deallocated at approximately the same time the corresponding entry is deallocated from the transaction table. In the case of DMIs/DIIs an entry in latency counter table is deallocated at approximately the same time corresponding response is received from the native interface. The deallocated entry counter value is then subtracted by the configured 8-bit offset, if the offset corrected value is negative then it is saturated downwards to zero. The quantizer takes the offset corrected value and increment appropriate histogram bin.

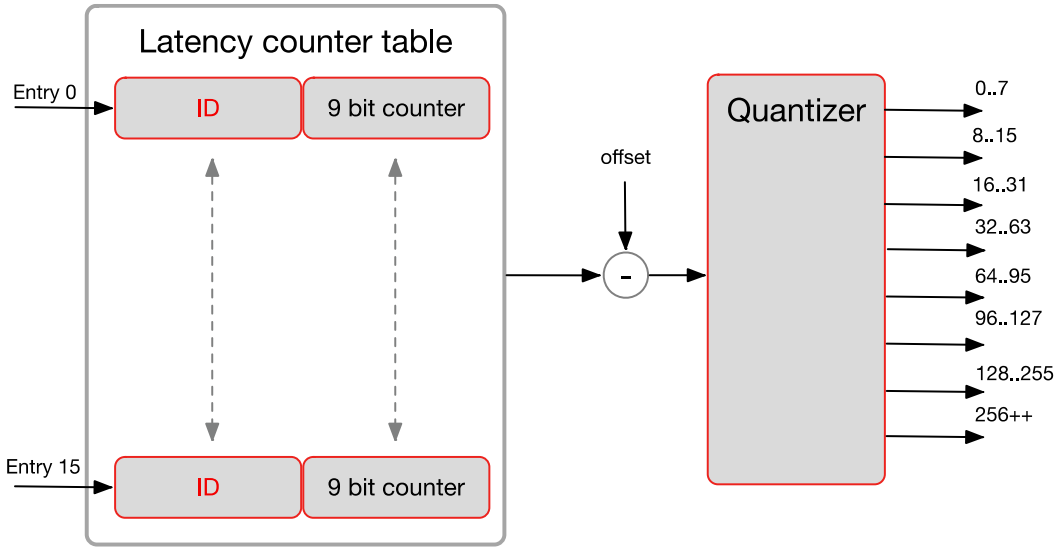


FIGURE 1 LATENCY HISTOGRAM BLOCK

To create a single histogram first 4 consecutive counter registers xCNTCR must be configured as follows

- ***Counter control*** to 3b100 i.e., bit bin counter
- ***SSR count*** to 3b100 i.e., 32 bit bin counter
- ***Count first event*** is don't care in this case
- ***Count Second event*** is don't care in this case

xLCNTCR must be configured as follows

- **Latency pre scale** to desired pre scale value
- **Read/Write latency** to desired read or write latency to be counted
- **Latency bin offset** to desired offset value
- **Latency count enable** to enable latency counting

xMCNTCR must be configured as follows

- ***Local Count Clear*** to 1b1 and clear the all local counters (this will clear all local counters other than the histogram counters)
- ***Local Count Enable*** to 1b1 and enable all local counters

After elapsed number of transactions or time or set ***Local Count Enable*** to 1b0 to stop the counters

8 histogram bins can be read from the 4 consecutive counter registers:

- ***xCNTVR0***: Bin 0

- ***xCNTSR0***: Bin 1
- ***xCNTVR1***: Bin 2
- ***xCNTSR1***: Bin 3
- ***xCNTVR2***: Bin 4
- ***xCNTSR2***: Bin 5
- ***xCNTVR3***: Bin 6
- ***xCNTSR3***: Bin 7

2 Opens

Questions/Feedback/Need to discuss:

3 Glossary

Arteris

A NoC Company

NCore3

A coherent NoC provided by Arteris with AMBA interfaces and built-in caches.

4 Notes

Notes .....