Ncore 3.4 - Architecture Parameter Documentation

Release: 3.4

Rev: 0.6 , March 24, 2023

ARTERIS® NCORE 3.4 - ARCHITECTURE PARAMETER DOCUMENTATION

Release Information

Version	Editor	Change	Date
0.1	MF/MK	Initial Document template created	10/24/2019
0.2	MF	Document started from template	01/26/2022
0.5	MF	Added details for each item and cleaned up	02/02/2022
0.6	CCW	Initial realease for Ncore 3.4	03/24/2023
Legend:	MK	Mohammed Khaleeluddin	
	MF	Michael Frank	
	CCW	Cheng Chung Wang	
	Xx	Whoever else edited this document	

Note:

- The initial description of these modifications has been presented as CCB request on Jan 21, 2022

Issues to be discussed:

- Do we want to apply this mechanism to write data buffers within DMI? In this case we would create a ***nCohWriteBufSiz*** parameter for each DMI and use a configuration register in each DMI to distribute them among the DCE
-

NCR32-PRD-11 CHI Protocol Support

Product implementation shall support CHI issue A and issue B features.

The implementation does not support following features

1. Retry mechanism
2. Write stashes from CHI B interface
3. CHI data check and power down mechanisms

ID	NCR32-PRD-11	
Name	CHI Protocol Support	
Status	Approved	
Release	3.0.0	
Owner	Mohammed Khaleeluddin	
Justification		
Notes	CHI base plus Stashing, Atomics, ROCI, ROMI, Prefetch.	

NCR32-PRD-12 ACE-Lite-E AIU Support

The implementation shall support ACE-Lite E compliance protocol by providing an ACE-Lite-E AIU.

Following limitations apply

1. Narrow accesses larger than 1 beat not supported
2. ACE5-Lite trace / loop back capability no supported
3. ACE5-lite power signaling, trace and loopback not supported

ID	NCR32-PRD-12	
Name	ACE-Lite-E AIU Support	
Status	Approved	
Release	3.0.0	
Owner	Mohammed Khaleeluddin	
Justification		
Notes		

NCR32-PRD-13 AXI AIU with Proxy Cache

The implementation shall support an AXI AIU with configurable Proxy Cache. MOESI caching model shall be supported.

ID	NCR32-PRD-13	
Name	AXI AIU with Proxy Cache	
Status	Approved	
Release	3.0.0	
Owner	Mohammed Khaleeluddin	
Justification		
Notes		

NCR32-PRD-14 DII and internal non-coherent connectivity

The product shall implement DII and internal non-coherent connectivity for handling of non-atomic, non-coherent IO transactions in the Ncore3 system

ID	NCR32-PRD-14	
Name	DII and internal non-coherent connectivity	
Status	Approved	
Release	3.0.0	
Owner	Mohammed Khaleeluddin	
Justification		
Notes		

NCR32-PRD-15 Support of Flexible Coherent Memory Map

Product implementation shall provide user ability to configure memory maps at system boot time.

ID	NCR32-PRD-15	
Name	Support of Flexible Coherent Memory Map	
Status	Approved	
Release	3.0.0	
Owner	Mohammed Khaleeluddin	
Justification		
Notes		

NCR32-PRD-16 Frequency of 1.6GHz

Product shall support most Ncore units running at 1.6 GHz, SS LVT/ULVT mix, 7nm.

The CHI AIU will have an optional CHI async interface capable of running up to 2GHz

ID	NCR32-PRD-16	
Name	Frequency of 1.6GHz	
Status	Approved	
Release	3.0.0	
Owner	Mohammed Khaleeluddin	
Justification		
Notes		

NCR32-PRD-17 Power Management features

Product shall implement power management features supporting 2 levels of clock-gating and ARM Q-channel support

ID	NCR32-PRD-17	
Name	Power Management features	
Status	Approved	
Release	3.0.0	
Owner	Mohammed Khaleeluddin	
Justification		
Notes		

NCR32-PRD-19 Debug and Trace Features

Product implementation shall provide debug and trace features at least equivalent to what is available in Ncore 2 while allowing for future compliance with Arm CoreSight.

ID	NCR32-PRD-19	
Name	Debug and Trace Features	
Status	Approved	
Release	3.2.0	
Owner	Mohammed Khaleeluddin	
Justification		
Notes		

NCR32-PRD-20 Performance Monitoring

Product shall implement baseline performance monitoring framework and capabilities such as performance counters and allow for expansion across multiple 3.x releases.

In release 3.2, Ncore is to come with additional performance counters at the Units levl (DCE, DVE, IOAIU, CHIAIU, DMI, DII blocks).

ID	NCR32-PRD-20	
Name	Performance Monitoring	
Status	Approved	
Release	3.2.0	
Owner	Mohammed Khaleeluddin	
Justification	NXP requesting the feature.	
Notes		

NCR32-PRD-23 CCP Capabilities into Ncore

Product implementation shall enable scratchpad and way partitioning for DMI.

ID	NCR32-PRD-23	
Name	CCP Capabilities into Ncore	
Status	Approved	
Release	3.0.0	
Owner	Mohammed Khaleeluddin	
Justification		
Notes		

NCR32-PRD-24 Read Acceleration

Product shall implement DTR read acceleration

ID	NCR32-PRD-24	
Name	Read Acceleration	
Status	Approved	
Release	3.0.0	
Owner	Mohammed Khaleeluddin	
Justification		
Notes		

NCR32-PRD-27 Support for different data widths

Implementation shall provide support for different data widths at the Ncore unit level

CHI AIU interface - 128/256

ACE/AXI/ACE-Lite AIU - 64/128/256

DII - 64/128/256

DMI - 128/256

ID	NCR32-PRD-27	
Name	Support for different data widths	
Status	Approved	
Release	3.0.0	
Owner	Mohammed Khaleeluddin	
Justification		
Notes	To see if there is any limited configuration we can restrict to	

NCR32-PRD-28 Separate out MN

Implementation shall separate out MN (Sys CSRs + DVM) from DCE

ID	NCR32-PRD-28	
Name	Separate out MN	
Status	Approved	
Release	3.0.0	
Owner	Mohammed Khaleeluddin	
Justification		
Notes	Assumption is that there is a top level "DCE0" wrapper that will include DCE0 + MN and that's what will be tested in a DCE unit level testbench	

NCR32-PRD-29 Ncore Security features

Product shall support Arm TrustZone

ID	NCR32-PRD-29	
Name	Ncore Security features	
Status	Approved	
Release	3.0.0	
Owner	Mohammed Khaleeluddin	
Justification		
Notes		

NCR32-PRD-202 Increase CAIU, IOAIU & DMI outstanding transactions amount

The CAIU, IOAIU & DMI interfaces are to be modified to support up to 128 (instead of 64 currently) outstanding transactions.

The DMI RTT limit is to also be increased to 128 (from 64) while the WTT limit is to remain at 64.

ID	NCR32-PRD-202	
Name	Increase CAIU, IOAIU & DMI outstanding transactions amount	
Status	Approved	
Release	3.2.0	
Owner	Mohammed Khaleeluddin	
Justification	NXP requested Ncore 3 to support up to 128 outstanding transactions on the CAIU, NCAIU and DMI interfaces. This is to ensure it covers the larger DRAM round-trip latency.	
Notes		

NCR32-PRD-203 PCIe ordering changes

Address manager changes

IOAIU changes: Support for updated GPRAs and ordering

DII changes: Changes to make sure writes are not blocked by reads

ID	NCR32-PRD-203	
Name	PCIe ordering changes	
Status	Approved	
Release	3.2.0	
Owner	Mohammed Khaleeluddin	
Justification	NXP	
Notes		

NCR32-PRD-204 Memory generic interface support

Ncore shall expose the memory functional ports (address, data, enables, etc.) at its top-level interface instead of instantiation the memories within the blocks where they are used.

ID	NCR32-PRD-204	
Name	Memory generic interface support	
Status	Approved	
Release	3.2.0	
Owner	Mohammed Khaleeluddin	
Justification	NXP requested this enhancement to ease the integration of the memories at the SoC top-level.	
Notes		

NCR32-PRD-205 Coherency requests support for power down

In order to support power down capabilities and to take peripherals in and out of coherence, messages need to be added (AMBA ACE5 SYSCOREQ and SYSCOACK signals at CAIU).

ID	NCR32-PRD-205	
Name	Coherency requests support for power down	
Status	Approved	
Release	3.2.0	
Owner	Mohammed Khaleeluddin	
Justification		
Notes		

NCR32-PRD-206 Exclusive events reporting

Ncore to support exclusive event reporting.

- Each AIU shall support a bi-directional event interface
- The event interface may receive events from the attached agent following the REQ/ACK-protocol
- The event interface shall forward events, generated internally or received from another agent, to the attached agent
- Events will be transported within Ncore using SysMsg transactions
- Events may originate within Ncore (when an exclusive monitor is cleared)
- Events entering Ncore at any AIU shall be forwarded to all AIUs

ID	NCR32-PRD-206	
Name	Exclusive events reporting	
Status	Approved	
Release	3.2.0	
Owner	Mohammed Khaleeluddin	
Justification	NXP is leveraging exclusive accesses.	
Notes		

NCR32-PRD-208 Error flow clean up

Enhanced Error flow implementation

ID	NCR32-PRD-208	
Name	Error flow clean up	
Status	Approved	
Release	3.2.0	
Owner	Mohammed Khaleeluddin	
Justification	NXP	
Notes		

NCR32-PRD-212 Increase of CHI and AXI user bit limit

Ncore to accept configurations with a 32 user bit limit for AXI and CHI, instead of 16.

ID	NCR32-PRD-212	
Name	Increase of CHI and AXI user bit limit	
Status	Approved	
Release	3.2.0	
Owner	Mohammed Khaleeluddin	
Justification	NXP requested this enhancement.	
Notes		

NCR32-PRD-213 SRAM address protection

SRAM addresses should be protected with an ECC.

ID	NCR32-PRD-213	
Name	SRAM address protection	
Status	Approved	
Release	3.2.0	
Owner	Mohammed Khaleeluddin	
Justification	This is a generic requirement which derives from the functional safety objectives.	
Notes		

NCR32-PRD-214 Limiting CSR access to one agent

Ncore should limit the access to the control and status registers (CSR) to one single agent to increase system robustness.

ID	NCR32-PRD-214	
Name	Limiting CSR access to one agent	
Status	Approved	
Release	3.2.0	
Owner	Mohammed Khaleeluddin	
Justification	NXP requested the enhancement for system robustness.	
Notes		

NCR32-PRD-215 QoS support

1. A QoS mechanism shall be implemented
2. The QoS mechanism shall accept the QoS or priority value supplied by the native agent protocol and map it to one of the corresponding service classes (bucket) of Ncore
3. The QoS mechanism shall support at least 8 different classes of service (QoS-buckets). These classes shall be encoded by a binary vector and propagated as part of the transaction metadata.
4. The QoS mechanism shall avoid priority inversion
5. The QoS mechanism shall ensure forward progress for all service classes (livelock/deadlock avoidance), one suggested method would be by implementing a starvation counter, limiting the number of higher service class transactions that may bypass any transaction of lower service class
6. Each service class shall determine the arbitration priority and a latency- or starvation count limit
7. A programmable threshold sets the starvation count limit, the max. number of higher service class transactions allowed to bypass any lower service class transaction, for each arbitration point - threshold values may be shared among one or more arbitration points
8. Arbitration priority shall be used at any point in the system where requests are queued up for processing, this includes but may not be limited to:
    a. Skid buffers (entry queue for all units)
    b. Transaction tables (AIU, DCE, DMI)
    c. Switches in the interconnect
9. A transaction's service class shall be upgraded after the threshold has been exceeded

ID	NCR32-PRD-215	
Name	QoS support	
Status	Approved	
Release	3.0.0	
Owner	Mohammed Khaleeluddin	
Justification		
Notes		

Confidential Proprietary Notice

This document is CONFIDENTIAL AND PROPRIETARY to Arteris, Inc. or its applicable subsidiary or affiliate (collectively or as applicable, "Arteris" or "Arteris IP"), and any use by you is subject to the terms of the agreement between you and Arteris IP or the terms of the agreement between you and the party authorized by Arteris IP to disclose this document to you.

This document is also protected by copyright and other related rights and the practice or implementation of the information contained in this document may be protected by one or more patents or pending patent applications. No part of this document may be reproduced in any form by any means without the express prior written permission of Arteris IP. **No license, express or implied, by estoppel or otherwise to any intellectual property rights is granted by this document unless specifically stated. You are prohibited from altering or deleting this notice from any use by you of this document.**

Your access to the information in this document is conditional upon your acceptance that you will not use or permit others to use the information: (i) for the purposes of determining whether implementations infringe any third party patents;(ii) for developing technology or products which avoid any of Arteris IP's intellectual property; or(iii) as a reference for modifying existing patents or patent applications or creating any continuation, continuation in part, or 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.

The Arteris IP name and corporate logo, and words marked with ® or ™ are registered trademarks or trademarks of Arteris (or its subsidiaries) in the US and/or elsewhere. All rights reserved. Other brands and names mentioned in this document may be the trademarks of their respective owners. Please follow Arteris IP's trademark usage guidelines, available from Arteris IP upon request by emailing to contracts@arteris.com.

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

1. *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.

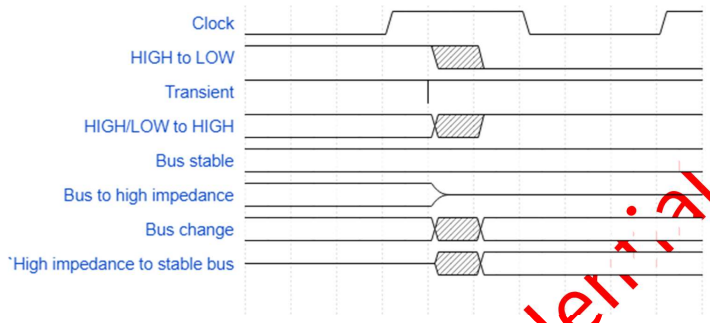
SMALL CAPITALS

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.

Definitions

Flow

Communication between two end points in the protocol. Includes sending a message from the initiator of a transaction (sender), for example an AIU, to the completer of the transaction (receiver), and returning a response back.

Target

The endpoint of a flow, Ncore architecture implements the following targets:

- DCE, DCE - commands from CAIU, NCAIU
- DMI - commands from CAIU, NCAIU and DCE; data from CAIU, NCAIU
- DII - commands & data from CAIU, NCAIU
- CAIU - snoop commands from DCE

2. *Overview*

This document describes parameters which are related with RTL and DV implementation. Parameters that are software cetric will be described in another document.

The main purpose of this document is to enumerate parameter for NCore 3.4. Therefore, several parameters for future purpose could be omitted.

3. *Assumptions*

NCore 3.0 has three parameter categories:

- pre-map parameters,
- post-map parameters which are being defined in Maestro, and
- hw-cpr files which are being defined in CPR file, mainly by HW design team.

Premap parameters are being used at Maestro mapping stage, and then post-map parameters override the values after the mapping. Finally, hw-cpr files will be used on top of results of software, and main purpose of this file is to define derivation rules from pre-map/post-map software-type files. This document is only summarizing preamp and postmap parameters of Maestro.

However, this document does not specify pre-map/post-map parameters. Instead, divide the parameters into (1) user settable parameters and (2) derived/fixed parameters. User parameter part will be visible to the customer through tcl configuration and GUI configuration. The default, min, and max value of the user settable parameters must match with user settable ranges.

3.1. System Constraints

System constraints do **NOT** correspond any user settable parameter. It is to add checks so that user cannot add more components over the limits.

TABLE 3-1: NUMBER OF CAIUS

Number of CAIUs	Architecture		Release		Default
	Min	*Max*	*Min*	*Max*	
Value			1	32	
Constraints					
Customer Description	Number of CAIUs				
Engineering Description					

TABLE 3-2: NUMBER OF NCAIUS

Number of NCAIUs	Architecture		Release		Default
	Min	*Max*	*Min*	*Max*	
Value			0	32	
Constraints					
Customer Description	Number of NCAIUs				
Engineering Description					

TABLE 3-3: NUMBER OF DMIS

Number of DMIs	Architecture		Release		Default
	Min	*Max*	*Min*	*Max*	.
Value			1	16	
Constraints					
Customer Description	Number of DMIs				
Engineering Description					

TABLE 3-4: NUMBER OF SFS

Number of SFs	Architecture		Release		Default
	Min	*Max*	*Min*	*Max*	
Value			1	16	
Constraints					
Customer Description	Number of Snoop Filters				
Engineering Description					

TABLE 3-5: NUMBER OF DVES

Number of DVEs	Architecture		Release		Default
	Min	*Max*	*Min*	*Max*	
Value			1	1	
Constraints					
Customer Description	Number of DVEs				
Engineering Description					

4. System User Settable Parameters

4.1. Project name parameters

TABLE 4-1: PROJECTNAME PARAMETER

Name: projectName				Visibility: User
	Architecture		**Release**	**Default**
	String		*String*	
Value				
Constraints				
Customer Description	Project Name.			
Engineering Description				

4.2. System level connectivity parameters

As described in Chapter 2.4 Message connectivity and network mapping of NCore3.2System spec, NCore provides the mapping templates of Concerto C messages to CDTI network. User will choose one of them considering the tradeoff between performance and area/power dissipation. For the detail of each mapping, refer NCore 3.2 System Architecture document.

We are supporting two options:

- Use two command networks and one data network
- Use three command networks and one data network

TABLE 4-2: COHERENTTEMPLATE PARAMETER

Name: coherentTemplate		Type: Enum	Visibiltiy: User
	Architecture	**Release**	**Default**
	Enum	*Enum*	
Value	'"TwoCtrlOneDataTemplate"', '"ThreeCtrlOneDataTemplate'		ThreeCtrlOne DataTemplate
Constraints			
Customer Description	Control and data network options: TwoCtrlOneDataTemplate: Adds support for two control and a single data network. ThreeCtrlOneDataTemplate: Adds support for three control and a single data network.		
Engineering Description			

New parameters are introduced to specify ports interleaving and to report connectivity information to AIUs.

TABLE 4-3: NAIUPORTS PARAMETER

Name: nAiuPorts			Type: Int		Visibiltiy: User
	Architecture		**Release**		**Default**
	Min	*Max*	*Min*	*Max*	
Value	1	16	1	4	1
Constraints	Powers of two valid values are 1, 2, 4, 8 and 16; Ports need to be same				
Customer Description	Specifies the number of AIU that are grouped together. These AIUs must be identical.				
Engineering Description	The parameter applies to any Initiator AIU type in Ncore i.e. CAIU, NCAIU or multi ported NCAIU These set of AIUs are treated as a single group of AIUs and must be identical. This parameter is on top of nNativeInterfacePorts as shown in Figure 4-1, here it shows as a mutliported NCAIU with two AXI ports specified by nNativeInterfacePorts and then 2 NCAIUs specified by nAiuPorts.				

TABLE 4-4: APRIMARYAIUPORTBITS PARAMETER

Name: aPrimaryAiuPortBits		Type: Enum	Visibiltiy: User
	Architecture	**Release**	**Default**
Value			array of integers
Constraints	Powers of two valid values are 1, 2, 4, 8 and 16; Ports need to be same		
Customer Description	Specifies the number of AIU that are grouped together. These AIUs must be identical.		
Engineering Description	The parameter applies to any Initiator AIU type in Ncore i.e. CAIU, NCAIU or multi ported NCAIU These set of AIUs are treated as a single group of AIUs and must be identical. This parameter is on top of nNativeInterfacePorts as shown in Figure 4-1, here it shows as a mutliported NCAIU with two AXI ports specified by nNativeInterfacePorts and then 2 NCAIUs specified by nAiuPorts.		

TABLE 4-5: ASECONDARYAIUPORTBITS PARAMETER

Name: aSecondaryAiuPortBits		Type: Enum	Visibiltiy: User
	Architecture	**Release**	**Default**
Value			Array of strings
Constraints	aSecondaryAiuPortBits is an array of string, its depth depends on nAiuPorts parameter value it is limited to log2(nAiuPorts). The string represents a hexadecimal number one hot encoded. Bits selected here cannot be same as the bits in aPrimaryAiuPortBits. Example aSecondaryAiuPortBits: [“'h4000”, “'h0”, “'h0”, “'h800”]		
Customer Description			
Engineering Description	Note used in this release		

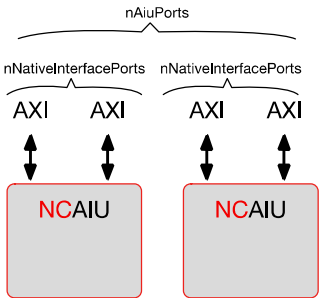


Figure 4-1 nAiuPorts Parameter

4.3. System address map parameters

At NCore 3.0, we could have up to 24 configurable memory regions, and each memory region would be configured using registers. Please refer system architecture spec for the detail (Chapter 3.3.2.12 and Chapter 3.3.2.13 General Purpose System Address Region Mapping Registers).

TABLE 4-6: NGPRA PARAMETER

Name: nGPRA			Type: int		Visibiltiy: User
	Architecture		**Release**		**Default**
	Min	*Max*	*Min*	*Max*	
Value	1	24			1
Constraints					
Customer Description	Number of general purpose address regions that the system can support				
Engineering Description					

DCE is supporting fixed style interleaving, and the interleaving bits are configurable using the above parameters.

TABLE 4-7: DCEINTERLEAVINGPRIMARYBITS

Name: dceInterleavingPrimaryBits		Type: Int	Visibiltiy: User
	Architecture	**Release**	**Default**
	Array of Integers	*Array of Integers*	
Value			
Constraints			
Customer Description	System directory primary select bits. N address bits other than bits 0 through 5 can be chosen. The cardinal values of these bits in the order of their ordinal positions are used to identify the DMIs to be accessed		
Engineering Description			

TABLE 4-8: DCEINTERLEAVINGSECONDARYBITS

dceInterleavingSecondaryBits		Type: Int	Visibility: None
	Architecture	**Release**	**Default**
	Array of Integers	*Array of Integers*	
Value		Will not be released at NCore 3.2	
Constraints			
Customer Description	The secondary bits are chosen on oa per primary bit bases. The bits within the set for a primary bit are combined and the primary bit with an Exclusive OR combination.		
Engineering Description	Not tested yet. Will not be released at NCore 3.2.		

4.4. System resiliency parameters

The resiliency feature in Ncore is optional, and when enabled, is implemented in addition to other configured Ncore features. The detail is described in Chapter 11 Resiliency Support of the NCoreSystem spec.

TABLE 4-9: RESILIENCY ON/OFF PARAMETER

Name: resiliencyEnabled		Type: Boolean	Visibility: User
	Architecture	**Release**	**Default**
	Boolean	*Boolean*	
Value	True, False		FALSE
Constraints			
Customer Description	Enable resilience-related features in the Ncore system.		
Engineering Description			

TABLE 4-10: DUPLICATION ENABLE PARAMETER

Name: duplicationEnabled		Type: Boolean	Visibility: User
	Architecture	**Release**	**Default**
	Boolean	*Boolean*	
Value	True, False		FALSE
Constraints			
Customer Description	Enable unit duplication for all Ncore units only. Memories and interconnect logic are not duplicated; they may be protected separately		
Engineering Description			

This capability enables a designer to source or terminate data protection signals on selected external CAIU, IO-AIU, DMI, and DII interfaces.

TABLE 4-11: NATIVE INTERFACE PROTECTION PARAMETER

Name: nativeIntfProtEnabled		Type: Boolean	Visibility: User
	Architecture	**Release**	**Default**
	Boolean	*Boolean*	
Value	True, False		FALSE
Constraints			
Customer Description	Enable capability to add protection on native Ncore interfaces. This adds an empty Verilog module with specified signals at the interface. Protection logic can be added in this Verilog module.		
Engineering Description			

The checker component receives one to four cycles delayed version of the same inputs as the functional component, which is decided by this parameter. The safety checker module receives the functional component outputs and delays them by one to four cycles, then compares them with the checker component outputs. Any discrepancy is considered a fault. Faults detected are logged and reported to the fault controller as mission fault. Once detected, the fault will remain logged inside the checker component until a BIST sequence clears it.

TABLE 4-12: INTERUNITDELAY PARAMETER

Name: interUnitDelay			Type: Int		Visibility: User
	Architecture		**Release**		**Default**
	Min	*Max*	*Min*	*Max*	
Value	1	4			1
Constraints					
Customer Description	Delay between functional unit and delay unit. Delay can be specified in number of clock cycles.				
Engineering Description					

TABLE 4-13: RESILIENCYPROTECTIONTYPE PARAMETER

Name: resiliencyProtectionType			Visibility: User
	Architecture	**Release**	**Default**
	String	*String*	
Value	'"NONE"', '"PARITY"', '"SECDED"		None
Constraints			
Customer Description	Interconnect protection type. Both data and control header will be protected. Available options are: NONE: no protection. PARITY: Error detection, parity protection. SECDED: Single bit error correction and double bit error detection, ECC protection.		
Engineering Description	This parameter affects CDTI protection only.		

TABLE 4-14: FNDISABLERESILIENCYBISTDEBUGPIN PARAMETER

Name: fnDisableResiliencyBistDebugPin			Type: Int		Visibility: User
	Architecture		**Release**		**Default**
	Min	*Max*	*Min*	*Max*	
Value	0	1	0	1	0
Constraints					
Customer Description	When set removes BIST and trace & debug disable pin.				
Engineering Description	When set removes BIST and trace & debug disable pin.				

4.4. System error parameters

This parameter configures timeout counter size in each module. We have additional register to configure the maximum size at run time, and the run time value should be less than or equal to this parameter value.

TABLE 4-15: TIMEOUTTHRESHOLDPARAMETER

Name: timeoutThreshold					Visibility: User
	Architecture		**Release**		**Default**
	Min	*Max*	*Min*	*Max*	
Value	1	2147483647			16384
Constraints					
Customer Description	Time out threshold value. This specifies number of clock cycles within which a transaction must complete in an NCORE system. The value specified is at 4096 clock cycle granularity.				
Engineering Description					

TABLE 4-16: MEMORYPROTECTIONTYPE PARAMETER

Name: memoryProtectionType			Visibility: User
	Architecture	**Release**	**Default**
	String	*String*	
Value	'"NONE"', '"PARITY"', '"SECDED"		None
Constraints			
Customer Description	Protection type for all memories in the Ncore system. Available options are: NONE : no protection. PARITY : Error detection, parity protection. SECDED : Single bit error correction and double bit error detection, ECC protection. SRAM memory type does not support memoryProtectionType ==NONE. If the memory is configured as FLOP, then NONE is supported.		
Engineering Description			

4.5. System QoS parameters

This parameter would enable starvation and aging arbitration in the skid buffer or OTT entry in AIU, DCE, and DMI.

TABLE 4-17: QOSENABLED PARAMETER

Name: qosEnabled		Visibility: User
	Architecture	**Default**
	Boolean	
Value	True, False	False
Constraints		
Customer Description	Enable QoS support	
Engineering Description		

The 4-bit QoS value for an incoming native transaction is mapped to one of 8 QoS buckets (3-bit value priority field) using this parameter. The mapped value are being used for the QoS arbitration in skid buffer and OTT entries in AIU, DCE, and DMI.

TABLE 4-18: QOSMAP PARAMETER

Name: qosMap			Visibility: User
	Architecture	**Release**	**Default**
	List of String	*List of String*	
Value			[]
Constraints			
Customer Description	4 bit Native interface QoS value map to 3 bit priority. **Value 0 has the highest priority and value 7 has the lowest priority**.		
Engineering Description			

Ncore 3 implements a single global counter as a time reference for starvation detection. Once the counter reaches a programmable threshold, an overflow bit in all active entries is set and the counter restarts. All transactions which had the overflow bit set at the time of the counter expiration will be considered starved and will be scheduled ahead of all non-starved transactions.

TABLE 4-19: QOSEVENTTHRESHOLD PARAMETER

Name: qosEventThreshold					Visibility: User
	Architecture		**Release**		**Default**
	Min	*Max*	*Min*	*Max*	
Value	1	8192			16
Constraints					
Customer Description	QoS starvation threshold. Maximum number of high priority requests that can bypass a lower priority request.				
Engineering Description					

4.6. System level debug parameters

Trace accumulate block (which is accumulate traces from AIU, DMI, and DII) is present only in DVE and the main functionality is to accumulate incoming trace DTWs from different NCore capture units. The capture buffer is sized based on the parameter nMainTraceBufSize

For the trace entries, user could configure as SRAM using user interface.

TABLE 4-20: NMAINTRACEBUFSIZE PARAMETER

Name: nMainTraceBufSize					Visibility: User
	Architecture		**Release**		**Default**
	Min	*Max*	*Min*	*Max*	
Value	32	4096	32	1024	64
Constraints					
Customer Description	Number of trace entries in the buffer				
Engineering Description	Number Debug DTW entries the trace buffer can hold. The actual depth of the trace buffer may be larger depending on the data width for the design. Each debug DTW can be max 64 bytes.				

All AIUs in the NCore system including those that support trace signaling shall have the capability to initiate transaction tracing using internal CSRs. An incoming transaction on the interfaces is compared with the trace CSR settings, if there is a match the transaction is marked to be traced. Multiple number of CSR sets can be present as specified at build time by the parameter nTraceRegisters

TABLE 4-21: NTRACEREGISTERS PARAMETER

Name: nTraceRegisters					Visibility: User
	Architecture		**Release**		**Default**
	Min	*Max*	*Min*	*Max*	
Value	1	8	1	4	1
Constraints	Minmum 1 is required				
Customer Description	Number of trace trigger configuration register sets. Each set of register can enable a trace condition.				
Engineering Description					

All AIUs, DMIs and DIIs shall support trace capturing capability. The block snoops SMI interface and captures messages that have the TraceMe field set. The capture block has a capture buffer that is sized based on the parameter nUnitTraceBufSize, this parameter specifies the number of 64-byte entries in the buffer.

For the trace entries, user could configure as SRAM using user interface.

TABLE 4-22: NUNITTRACEBUFSIZES PARAMETER

Name: nMainTraceBufSize					Visibility: Engg
	Architecture		**Release**		**Default**
	Min	*Max*	*Min*	*Max*	
Value	8	32	8	16	8
Constraints	Allowable size are power of 2				
Customer Description	Number of trace entries in each Ncore Unit. Each entry is 64 bytes				
Engineering Description					

To enable debug of a hung Ncore system a slave APB port must be added to the CSR network that can access all the Ncore CSRs. At top level this port signals must be "<prefix>_debug_apb_<rest of the signal name>".
Following APB port restrictions apply

- Fixed data bus width 32 bits
- Fixed address bus width of 20 bits
- Fixed access size of 4 bytes
- All access are 4 byte aligned.

This port is expected to be used for debug only, if same register is accessed concurrently via this debug APB port and the internal Ncore CSR accesses then the effect on the CSR is undefined. Ncore does not guarantee any ordering between the two access.

TABLE 4-23: FNDEBUGAPBENABLE PARAMETER

Name: fnDebugAPBEnable					Visibility: User
	Architecture		**Release**		**Default**
	Min	*Max*	*Min*	*Max*	
Value	0	1	0	1	1
Constraints					
Customer Description	When set enables an APB slave port on the CSR network. This port is expected to be used for on chip debug purposes only.				
Engineering Description	When set enables an APB slave port on the CSR network. This port is expected to be used for on chip debug purposes only.				

4.7. System level physical parameters

TABLE 4-24: SYNCDEPTH PARAMETER

Name: syncDepth			Visibility: User
	Architecture	**Release**	**Default**
	Valid values	*Valid values*	
Value	2, 3, 4		2
Constraints			
Customer Description	The depth of the synchronizers used for signals that cross domains for metastability reasons. This is only for sym_async_adapter. FIFO depth of the chi_async_adapter would be calculate considering credit at the CHI interface link.		
Engineering Description	Circular FIFO depth of the sym_async_adpater would be derived by this system configuration value • syncDepth: 2 → circular fifo depth of sym_async_adapter: 8 • syncDepth: 3 → circular fifo depth of sym_async_adapter: 10 • syncDepth: 4 → circular fifo depth of sym_async_adapter: 12		

4.8. System engineering parameters

Engineering Only parameter should not be visible to the customer.

TABLE 4-25: ASSERTIONENABLE PARAMETER

Name: assertionEnable				Visibility: Engg
	Description	**Type**	**Default**	**Notes**
assertionEnable	Enable HW assertions	Boolean	FALSE	Engineering Only

5. *Power and Clock User Settable Parameters*

Clocking and Power are defined in terms of regions, domains, and sub domains:

- A power region represents a group of elements that run off a power supply that is driven by one power source.
- A clock region represents a group of elements that are clocked by single clock.
- A power domain represents a group of elements whose power can be turned on and off.
- A clock domain represents a group of elements whose clock can be turned on and off.
- There are no power sub domains.
- A clock sub domain is a group of elements in a clock domain whose clocks can be turned on and off dynamically as a part of logic function (clock divider). There are no clock dividers supported in Ncore 3.2.

FIGURE 5-1: POWER AND CLOCK DOMAIN DEFINITION FOR NCORE 3.2

Ncore 3.2 supports three levels of clock gating:

- The first level clock gating is enabled by synthesis tools, for example Synopsys Design Compiler.
- A second level of clock gating will be inserted, one per Ncore unit. This level gates the clock for the complete unit when no active transactions are within that unit. This is enabled by "unitClockGating" parameter of clock region.
- A third level of clock gating can be achieved by using the q-channel.

This is enabled by "Gating" parameter of clock domain.

How to achieve third-level clock gating scheme is under discussion https://jira.arteris.com/browse/MAES-3988. The below is proposal from John/MK.

At NCore 3.2, we would want to support (only) **NCore unit clock gating** using q-channel. To achieve this, the below updates would be required:

- By default, clock subdomains are async with other clock domains, but NCore 3.2 would need to allow clock domains which share the same clock root to be explicitly defined to be synchronous with each other.
- The reason is to allow the Ncore units to be gated without gating the CSR network and potentially other networks so that those messages do not get accidentally trapped. In this case user would define two clock domains which were synchronous to each other, with one of them being dynamic and the other not. The dynamic clock would be used for connecting the Ncore units while the non-dynamic clock would be associated with other components such as the CSR network.

To support the above, the following changes would be needed:

- Sub domain update:
  - There will be one single clock subdomain per clock domain. There are no clock dividers.
  - Only allow one clock subdomain per domain
- Clock domain update:
  - Will allow clock domains which share the same clock root and explicitly defined to be synchronous.
  - Async adapter would not be inserted at synchronous clock boundary.

NCore 3.2 restrictions:

- NCore 3.2 supports only single power domain.
  - NCore 3.2 does NOT provide a UPF file to the customer that describes where the level shifters and clamping cells (and the associated clamping values) for signals that cross between power domains.
  - It is user's reponsiblity to support multiple power domain using clock region/domain capabiltiy.
- NCore 3.2 does NOT support retention mode.
- NCore 3.2 does NOT support auto-wakeup, that is, QACTIVE during the powered down state will not assert indicating a request to the PMU (User's power control unit) to wake up.

Detach process (SysCoReq/SysCoAck):

Before NCore unit clock gating, attach and Detach to the coherent domain should be performed by SysCo/SysAck. This will be controlled by CPU events or CSR interface setting.

TABLE 3.2: PARAMETER RELATED WITH **CLOCK REGION**: FREQUENCY

Name: Frequency					Visibility: User
	Architecture		**Release**		**Default**
	Min	*Max*	*Min*	*Max*	
Value	1	160000000			300000
Constraints					
Customer Description					
Engineering Description	NCore 3.2 supports 1.6GHz as a maximum frequency. The range should visible only for that ranges.				

TABLE 5-2: PARAMETER RELATED WITH **CLOCK REGION:** UNITCLOCKGATING

Name: unitClockGating			Visibility: User
	Architecture	**Release**	**Default**
	Valid Values	*Valid values*	
Value	True, False		FALSE
Constraints			
Customer Description	When the parameter is true, then the blocks in the corresponding clock region will insert clock gating based on its internal and the state of the interfaces connected to it.		
Engineering Description	Not all blocks will insert clock gates when this parameter is set to true. For instance, blocks sym_async_adapter and sym_rate_adapter do not insert clock gating in response to this parameter.		

TABLE 5-3: PARAMETER RELATED WITH **CLOCK DOMAIN:** GATING

Name: Gating			Visibility: User
	Architecture	**Release**	**Default**
	Valid Values	*Valid Values*	
Value	always_on, external		always_on
Constraints			
Customer Description	Specify 'always_on' if no gating applied, or 'external' if logic externa		
Engineering Description			

6. *Memory Map User Settable Parameters*

Ncore 3.2 address map is categorized into three main spaces:

- Ncore Register Space (NRS): This address space is reserved by Ncore 3 architecture for mapping Control and Status registers belonging to Ncore 3 units. Each Ncore 3 unit's registers map within a single 4 KB block of address space.
- General Purpose Address Space (GPAS): The remaining address space is available for general purpose use. It may contain multiple system memory or peripheral storage ranges. General purpose address space may be comprised of one or regions of type system memory or peripheral storage. The system memory regions can be accessed coherently or non-coherently.
- Boot Region (BR): Ncore 3 permits the SoC system to identify a contiguous aligned block of address space for the boot code to reside in. The boot code might be accessed by a processor during the system boot process when no other address mapping might be valid. The type of storage occupied by the Boot Space can be system memory or peripheral memory.

Listed below are parameters used to configurable memory space:

TABLE 6-1: PARAMETER RELATED WITH **CSR REGION**: MEMORYBASE

Name: memoryBase			Visibility: User
	Architecture	**Release**	**Default**
	Valid Values	*Valid values*	
Value			0x0
Constraints			
Customer Description	Specify CSR region base address. This address must be aligned to the size specified.		
Engineering Description			

TABLE 6-2: PARAMETER RELATED WITH **CSR REGION**: MEMORYSIZE

Name: memorySize			Visibility: User
	Architecture	**Release**	**Default**
	Valid Values	*Valid values*	
Value	1MB	1MB	1MB
Constraints			
Customer Description	CSR sized is fixed as 1MB at NCore 3.2		
Engineering Description			

TABLE 6-3: PARAMETER RELATED WITH **BOOT REGION**: MEMORYBASE

Name: memoryBase			Visibility: User
	Architecture	**Release**	**Default**
	Valid Values	*Valid values*	
Value			0x0
Constraints			
Customer Description	Specify boot region base address. This address must be aligned to the size specified.		
Engineering Description	Must be aligned to 4KB		

TABLE 6-4: PARAMETER RELATED WITH **BOOT REGION**: MEMORYSIZE

Name: memorySize			Visibility: User
	Architecture	**Release**	**Default**
	Valid Values	*Valid values*	
Value	Min: 4KB, Max: 32KB		4K
Constraints			
Customer Description	Specifies the size of boot region. Minimum is 4KB and must be a power of two.		
Engineering Description			

TABLE 6-5: PARAMETER RELATED WITH **BOOT REGION**: MG_REF

Name: mg_ref			Visibility: User
	Architecture	**Release**	**Default**
	Valid Values	*Valid values*	
Value			
Constraints			
Customer Description	Specify the DMI interleave group associated with the boot region.		
Engineering Description	If mg_ref is specified, channel_ref cannot be specified		

TABLE 6-6: PARAMETER RELATED WITH **BOOT REGION**: CHANNEL_REF

Name: channel_ref			Visibility: User
	Architecture	**Release**	**Default**
	Valid Values	*Valid values*	
Value			
Constraints			
Customer Description	Specify the DII group associated with the boot region.		
Engineering Description	If channel_ref is specified, mg_ref cannot be specified		

DYNAMIC MEMORY GROUP

Dynamic memory group is to define GPARS (Number of GPARS is configured by Table 4-6: nGPRA parameter). For each dynamic memory group, the target DMIs are bounded. The base and size are configured for each group. If we bound more than two DMIs into one dynamic memory group, we need interleaving granularity, which is configured by Interleaving Functions.

- In NCore 3.2/3.4, user could bound only 1, 2, 4, 8, and 16 DMIs into one dynamic memory group (=MIG).
- If a dynamic memory group have more than 2 DMIs, we need to define the interleaving function (=MIF) for each DMI. The below tables are user settable parameters to define interleave function.
- The maximum number of interleaving functions (=interleaving granularity)is 2 in NCore 3.2/3.4.

Please see Chapter 3.3 Address Map Specification in System Architecture spec for the detail.

The primaryInterleavingBits are used to specify interleaving function per each interleaving group.

TABLE 6-7: PRIMARYINERLEAVINGBITONE

Name: primaryInterleavingBitOne			Type: Int		Visibility: User
	Architecture		Release		Default
	Min	*Max*	*Min*	*Max*	
Value	0	8192			0
Constraints					
Customer Description					
Engineering Description	This primaryInterleavingBitOne is per MIF.				

TABLE 6-8: PRIMARYINERLEAVINGBITTWO

Name: primaryInterleavingBitTwo			Type: Int		Visibility: User
	Architecture		Release		Default
	Min	*Max*	*Min*	*Max*	
Value	0	8192			0
Constraints					
Customer Description					
Engineering Description	This primaryInterleavingBitTwo is per MIF.				

TABLE 6-9: PRIMARYINERLEAVINGBITTHREE

Name: primaryInterleavingBitThree			Type: Int		Visibility: User
	Architecture		Release		Default
	Min	*Max*	*Min*	*Max*	
Value	0	8192			0
Constraints					
Customer Description					
Engineering Description	This primaryInterleavingBitThree is per MIF.				

TABLE 6-10: PRIMARYINERLEAVINGBITFOUR

Name: primaryInterleavingBitFour		Type: Int	Visibility: User
	Architecture	Release	Default
	Min	*Max*	*Min*
Value	0	8192	
Constraints			
Customer Description			
Engineering Description	This primaryInterleavingBitFour is per MIF.		

7. Socket User Settable Parameters

All the interfaces are defined based on AXI_Interface (except APB.) That is, AXI_interface parameters are defined first and each interface parameters will be defined on top of it. Will be overwritten if there is any duplicated parameters.

The parameter in this chapter is only for CDTI (Control and data transport interconnect). For the CSTI (Control and status transport interconnect) , all the parameters are fixed and described in Chapter 19.8

NCore 3.2 restrictions:

- AwID and ArID need to be restricted to be same for an interface irrespective of it being a master or slave.
- Header user bit: wArUser, wAwUser. These values per socket must be all the same and must be the same for all Sockets.
    - wArUser = wAwUser for all the sockets in the request and response network.
- All Sockets need to have the same address width.

7.1. AXI_Interface

TABLE 7-1: PARAMETER RELATED WITH AXI INTERFACE: WARID

Name: wArID					Visibility: User
	Architecture		**Release**		**Default**
	Min	*Max*	*Min*	*Max*	
Value	1	10/20			6
Constraints					
Customer Description	Specify the width of AXI interface ArId bits. AIU: [1:10], DMI/DII: [1:20]				
Engineering Description	There is an constraint between initator and target ID width. Target ID width must be equal or larger than (maximum of all the AxIDs and wLPId) + wFUnitId.				

TABLE 7-2: PARAMETER RELATED WITH AXI INTERFACE: WAWID

Name: wAwID					Visibility: User
	Architecture		**Release**		**Default**
	Min	*Max*	*Min*	*Max*	
Value	1	10/20			6
Constraints					
Customer Description	Specify the width of AXI interface AwId bits. AIU: [1:10], DMI/DII: [1:20]				
Engineering Description	There is an constraint between initator and target ID width. Target ID width must be equal or larger than (maximum of all the AxIDs and wLPId) + wFUnitId.				

TABLE 7-3: PARAMETER RELATED WITH AXI INTERFACE: WADDR

Name: wAddr					Visibility: User
	Architecture		**Release**		**Default**
	Min	*Max*	*Min*	*Max*	
Value	12	64			32
Constraints					
Customer Description	Specify the width of AXI interface address bits.				
Engineering Description					

TABLE 7-4: PARAMETER RELATED WITH AXI INTERFACE: WDATA

Name: wData			Visibility: User
	Architecture	**Release**	**Default**
	Valid values	*Valid values*	
Value	['32', '64', '128', '256'] • AIU - 64/128/256 • DII - 64/128/256 • ConfigDII: 32 • DMI - 128/256		AIU/DII: 64 DMI: 128
Constraints			
Customer Description	Specify the width of AXI interface data bits. Following limitations apply. AXI interface connected to memory as Ncore master(DMI): 128 & 256. AXI interface connected to peripheral device Ncore master (DII): 64, 128 & 256. AXI interface connected to a master agent accelerator, GPU, GIC etc. as Ncore slave (AIU): 64, 128 & 256.		
Engineering Description			

TABLE 7-5: PARAMETER RELATED WITH AXI INTERFACE: AWUSER

Name: wAwUser					Visibility: User
	Architecture		**Release**		**Default**
	Min	*Max*	*Min*	*Max*	
Value	0	32			0
Constraints					
Customer Description	Width of user bit on AW AXI Interface				
Engineering Description					

TABLE 7-6: PARAMETER RELATED WITH AXI INTERFACE: ARUSER

Name: wArUser					Visibility: User
	Architecture		**Release**		**Default**
	Min	*Max*	*Min*	*Max*	
Value	0	32			0
Constraints					
Customer Description	Width of user bit on AR AXI Interface				
Engineering Description					

7.2. ACE_Interface

TABLE 7-7: PARAMETER RELATED WITH ACE INTERFACE: WARID

Name: wArID					Visibility: User
	Architecture		**Release**		**Default**
	Min	*Max*	*Min*	*Max*	
Value	1	10			6
Constraints					
Customer Description	Specify the width of ACE interface ArId bits.				
Engineering Description					

TABLE 7-8: PARAMETER RELATED WITH ACE INTERFACE: WAWID

Name: wAwID					Visibility: User
	Architecture		**Release**		**Default**
	Min	*Max*	*Min*	*Max*	
Value	1	10			6
Constraints					
Customer Description	Specify the width of ACE interface AwId bits.				
Engineering Description					

TABLE 7-9: PARAMETER RELATED WITH ACE INTERFACE: WADDR

Name: wAddr				Visibility: User
	Architecture	**Release**		**Default**
	Valid Values			
Value	32, 40, 44, 48			32
Constraints				
Customer Description	Specify the width of ACE interface address bits. ACE only: 32, 40, 44, 48 ACE with CHI: 44 ACE with CHI-B: 44, 48			
Engineering Description				

TABLE 7-10: PARAMETER RELATED WITH ACE INTERFACE: WDATA

Name: wData			Visibility: User
	Architecture	**Release**	**Default**
	Valid values	*Valid values*	
Value	64, 128, 256		64
Constraints			
Customer Description	Specify the width of ACE interface data bits		
Engineering Description			

TABLE 7-11: PARAMETER RELATED WITH ACE INTERFACE: AWUSER

Name: wAwUser					Visibility: User
	Architecture		**Release**		**Default**
	Min	*Max*	*Min*	*Max*	
Value	0	32			0
Constraints					
Customer Description	Width of user bit on AW ACE Interface				
Engineering Description					

TABLE 7-12: PARAMETER RELATED WITH ACE INTERFACE: ARUSER

Name: wArUser					Visibility: User
	Architecture		**Release**		**Default**
	Min	*Max*	*Min*	*Max*	
Value	0	32			0
Constraints					
Customer Description	Width of user bit on AR ACE Interface				
Engineering Description					

TABLE 7-13: PARAMETER RELATED TO SYSCO INTERFACE

Name: useSysCoInt			Visibility: User
	Architecture	**Release**	**Default**
	Boolean	*Boolean*	
Value	*True, False*	*True, False*	True
Constraints	Default True to ACE/CHI interface CPU, but if customer's CPU does not have the SysCo interface, customers can set the parameter to be False.		
Customer Description	Setting the parameter to enable the connection of SysCoReq and SysCoAck interface to the AIU. if customer's CPU does not have the interface, and does not set False to the parameter, it is recommended to tie SysCoReq to 0.		
Engineering Description	Connect the I/O to the SysCo Engine hardware.		

TABLE 7-14: PARAMETER RELATED TO EVENTIN INTERFACE

Name: useEventInInt			Visibility: User
	Architecture	**Release**	**Default**
	Boolean	*Boolean*	
Value	*True, False*	*True, False*	True
Constraints	Default True to ACE/CHI interface CPU, but if customer's CPU does not have the EventIn interface, customers can set the parameter to be False.		
Customer Description	Setting the parameter to enable the connection of EventInReq and EventInAck interface to the AIU. if customer's CPU does not have the interface, and does not set False to the parameter, it is recommended to tie EventInAck to EventInReq.		
Engineering Description	Connect the I/O to the SysReq Receiver hardware.		

TABLE 7-15: PARAMETER RELATED TO EVENTOUT INTERFACE

Name: useEvenOutInt			Visibility: User
	Architecture	**Release**	**Default**
	Boolean	*Boolean*	
Value	*True, False*	*True, False*	True
Constraints	Default True to ACE/CHI interface CPU, but if customer's CPU does not have the EventOut interface, customers can set the parameter to be False.		
Customer Description	Setting the parameter to enable the connection of EventOutReq and EventOutAck interface to the AIU. if customer's CPU does not have the interface, and does not set False to the parameter, it is recommended to tie EventOutReq to 0.		
Engineering Description	Connect the I/O to the SysReq Sender hardware.		

7.3. ACE-LITE E_Interface

TABLE 7-16: PARAMETER RELATED WITH ACELITEE INTERFACE: WARID

Name: wArID					Visibility: User
	Architecture		**Release**		**Default**
	Min	*Max*	*Min*	*Max*	
Value	1	10			6
Constraints					
Customer Description	Specify the width of ACELITEE interface ARID bits.				
Engineering Description					

TABLE 7-17: PARAMETER RELATED WITH ACELITEE INTERFACE: WAWID

Name: wAwID					Visibility: User
	Architecture		**Release**		**Default**
	Min	*Max*	*Min*	*Max*	
Value	1	10			6
Constraints					
Customer Description	Specify the width of ACELITEE interface AwId bits.				
Engineering Description					

TABLE 7-18: PARAMETER RELATED WITH ACELITEE INTERFACE: WADDR

Name: wAddr					Visibility: User
	Architecture		**Release**		**Default**
	Min	*Max*	*Min*	*Max*	
Value	12	64			32
Constraints					
Customer Description	Specify the width of ACELITEE interface Address bits.				
Engineering Description					

TABLE 7-19: PARAMETER RELATED WITH ACELITEE INTERFACE: WDATA

Name: wData			Visibility: User
	Architecture	**Release**	**Default**
	Valid values	*Valid values*	
Value	64, 128, 256		64
Constraints			
Customer Description	Specify the width of ACELITE interface data bits		
Engineering Description			

TABLE 7-20: PARAMETER RELATED WITH ACELITEE INTERFACE: AWUSER

Name: wAwUser					Visibility: User
	Architecture		**Release**		**Default**
	Min	*Max*	*Min*	*Max*	
Value	0	32			0
Constraints					
Customer Description	Width of user bit on AW ACELITE Interface				
Engineering Description					

TABLE 7-21: PARAMETER RELATED WITH ACELITEE INTERFACE: ARUSER

Name: wArUser					Visibility: User
	Architecture		**Release**		**Default**
	Min	*Max*	*Min*	*Max*	
Value	0	32			0
Constraints					
Customer Description	Width of user bit on AR ACELITE Interface				
Engineering Description					

TABLE 7-22: PARAMETER RELATED WITH ACELITEE INTERFACE: EAC

Name: **eAC**			Visibility: User
	Architecture	**Release**	**Default**
	Valid values	*Valid values*	
Value	0, 1		0
Constraints			
Customer Description	Enable AC snoop bus to support DVM		
Engineering Description	Archi team would modify the parameter name after NCore 3.2.		

TABLE 7-23: PARAMETER RELATED TO EVENTOUT INTERFACE

Name: useEvenOutInt				Visibility: User
	Architecture		**Release**	**Default**
	Boolean		*Boolean*	
Value	*True, False*		*True, False*	False
Constraints	Default True to ACE/CHI interface CPU, but if customer's CPU does not have the EventOut interface, customers can set the parameter to be False.			
Customer Description	Setting the parameter to enable the connection of EventOutReq and EventOutAck interface to the AIU. if customer's CPU does not have the interface, and does not set False to the parameter, it is recommended to tie EventOutReq to 0.			
Engineering Description	Connect the I/O to the SysReq Sender hardware.			

7.4. ACE LITE Interface

TABLE 7-24: PARAMETER RELATED WITH ACELITE INTERFACE: WARID

Name: wArID					Visibility: User
	Architecture		**Release**		**Default**
	Min	*Max*	*Min*	*Max*	
Value	1	10			6
Constraints					
Customer Description	Specify the width of ACELITE interface ARID bits.				
Engineering Description					

TABLE 7-25: PARAMETER RELATED WITH ACELITE INTERFACE: WAWID

Name: wAwID					Visibility: User
	Architecture		**Release**		**Default**
	Min	*Max*	*Min*	*Max*	
Value	1	10			6
Constraints					
Customer Description	Specify the width of ACELITE interface AwId bits.				
Engineering Description					

TABLE 7-26: PARAMETER RELATED WITH ACELITE INTERFACE: WADDR

Name: wAddr					Visibility: User
	Architecture		**Release**		**Default**
	Min	*Max*	*Min*	*Max*	
Value	12	64			32
Constraints					
Customer Description	Specify the width of ACELITE interface Address bits.				
Engineering Description					

TABLE 7-27: PARAMETER RELATED WITH ACELITE INTERFACE: WDATA

Name: wData			Visibility: User
	Architecture	**Release**	**Default**
	Valid values	*Valid values*	
Value	64, 128, 256		64
Constraints			
Customer Description	Specify the width of ACELITE interface data bits		
Engineering Description			

TABLE 7-28: PARAMETER RELATED WITH ACELITE INTERFACE: AWUSER

Name: wAwUser					Visibility: User
	Architecture		**Release**		**Default**
	Min	*Max*	*Min*	*Max*	
Value	0	32			0
Constraints					
Customer Description	Width of user bit on AW ACELITE Interface				
Engineering Description					

TABLE 7-29: PARAMETER RELATED WITH ACELITE INTERFACE: ARUSER

Name: wArUser					Visibility: User
	Architecture		**Release**		**Default**
	Min	*Max*	*Min*	*Max*	
Value	0	32			0
Constraints					
Customer Description	Width of user bit on AR ACELITE Interface				
Engineering Description					

TABLE 7-30: PARAMETER RELATED WITH ACELITE INTERFACE: EAC

Name: **eAC**			Visibility: User
	Architecture	**Release**	**Default**
	Valid values	*Valid values*	
Value	0, 1		0
Constraints			
Customer Description	Enable AC snoop bus to support DVM		
Engineering Description	Archi team would modify the parameter name after NCore 3.2.		

TABLE 7-31: PARAMETER RELATED TO EVENTOUT INTERFACE

Name: useEvenOutInt			Visibility: User
	Architecture	**Release**	**Default**
	Boolean	*Boolean*	
Value	*True, False*	*True, False*	False
Constraints	Default True to ACE/CHI interface CPU, but if customer's CPU does not have the EventOut interface, customers can set the parameter to be False.		
Customer Description	Setting the parameter to enable the connection of EventOutReq and EventOutAck interface to the AIU. if customer's CPU does not have the interface, and does not set False to the parameter, it is recommended to tie EventOutReq to 0.		
Engineering Description	Connect the I/O to the SysReq Sender hardware.		

7.5. CHI Issue A Interface

TABLE 7-32: PARAMETER RELATED WITH CHI_A_INTERFACE: WADDR

Name: wAddr			Visibility: User
	Architecture	**Release**	**Default**
	Valid Values	*Valid values*	
Value	44		44
Constraints			
Customer Description	Addr width of CHI interface		
Engineering Description			

TABLE 7-33: PARAMETER RELATED WITH CHI_A_INTERFACE: WDATA

Name: wData			Visibility: User
	Architecture	**Release**	**Default**
	Valid Values	*Valid values*	
Value	128, 256	128, 256	
Constraints			
Customer Description	Width of data on the CHI interface		
Engineering Description			

TABLE 7-34: PARAMETER RELATED WITH CHI_A_INTERFACE: REQ_RSVDC

Name: REQ_RSVDC			Visibility: User
	Architecture	**Release**	**Default**
	Valid Values	*Valid values*	
Value	['0', '4', '8', '12', '16', '24', '32']	['0', '4', '8', '12', '16', '24', '32']	
Constraints			
Customer Description	REQ_RSVDC is to define user-bit for command channel. Do not support user bit on data channel.		
Engineering Description			

TABLE 7-35: PARAMETER RELATED TO SYSCO INTERFACE

Name: useSysCoInt			Visibility: User
	Architecture	**Release**	**Default**
	Boolean	*Boolean*	
Value	*True, False*	*True, False*	True
Constraints	Default True to ACE/CHI interface CPU, but if customer's CPU does not have the SysCo interface, customers can set the parameter to be False.		
Customer Description	Setting the parameter to enable the connection of SysCoReq and SysCoAck interface to the AIU. if customer's CPU does not have the interface, and does not set False to the parameter, it is recommended to tie SysCoReq to 0.		
Engineering Description	Connect the I/O to the SysCo Engine hardware.		

TABLE 7-36: PARAMETER RELATED TO EVENTIN INTERFACE

Name: useEvenInInt			Visibility: User
	Architecture	**Release**	**Default**
	Boolean	*Boolean*	
Value	*True, False*	*True, False*	True
Constraints	Default True to ACE/CHI interface CPU, but if customer's CPU does not have the EventIn interface, customers can set the parameter to be False.		
Customer Description	Setting the parameter to enable the connection of EventInReq and EventInAck interface to the AIU. if customer's CPU does not have the interface, and does not set False to the parameter, it is recommended to tie EventInAck to EventInReq.		
Engineering Description	Connect the I/O to the SysReq Receiver hardware.		

TABLE 7-37: PARAMETER RELATED TO EVENTOUT INTERFACE

Name: useEvenOutInt			Visibility: User
	Architecture	**Release**	**Default**
	Boolean	*Boolean*	
Value	*True, False*	*True, False*	True
Constraints	Default True to ACE/CHI interface CPU, but if customer's CPU does not have the EventOut interface, customers can set the parameter to be False.		
Customer Description	Setting the parameter to enable the connection of EventOutReq and EventOutAck interface to the AIU. if customer's CPU does not have the interface, and does not set False to the parameter, it is recommended to tie EventOutReq to 0.		
Engineering Description	Connect the I/O to the SysReq Sender hardware.		

7.6. CHI Issue B Interface

what are the changes between A and B?

TABLE 7-38: PARAMETER RELATED WITH CHI_B_INTERFACE: NODEID_WIDTH

Name: NodeID_Width					Visibitity: User
	Architecture		**Release**		**Default**
	Min	Max	min	max	
Value	7	11			7
Constraints					
Customer Description	Width of the Node ID of the CHI Interface.				
Engineering Description					

TABLE 7-39: PARAMETER RELATED WITH CHI_B_INTERFACE: WADDR

Name: wAddr					Visibitity: User
	Architecture		Release		Default
	Min	Max	min	max	
Value	44	52			48
Constraints					
Customer Description	Width of the address on CHI interface.				
Engineering Description					

TABLE 7-40: PARAMETER RELATED WITH CHI_B_INTERFACE: REQ_RSVDC

Name: REQ_RSVDC			Visibitity: User
	Architecture	**Release**	**Default**
	Valid Values	*Valid Values*	
Value	['0', '4', '8', '12', '16', '24', '32']		0
Constraints			
Customer Description	REQ_RSVDC is to define user-bit for command channel. Do not support user bit on data channel.		
Engineering Description			

TABLE 7-41: PARAMETER RELATED WITH CHI_B_INTERFACE: WDATA

Name: wData			Visibitity: User
	Architecture	**Release**	**Default**
	Valid Values	*Valid Values*	
Value	128, 256		128
Constraints			
Customer Description	Width of data on the Chi interface		
Engineering Description			

TABLE 7-42: PARAMETER RELATED WITH CHI_B_INTERFACE: ENPOISON

Name: enPoison			Visibitity: User
	Architecture	**Release**	**Default**
	Valid Values	*Valid Values*	
Value	True, False		False
Constraints			
Customer Description	Enable Poison Bit		
Engineering Description			

TABLE 7-43: PARAMETER RELATED TO SYSCO INTERFACE

Name: useSysCoInt			Visibility: User
	Architecture	**Release**	**Default**
	Boolean	*Boolean*	
Value	*True, False*	*True, False*	True
Constraints	Default True to ACE/CHI interface CPU, but if customer's CPU does not have the SysCo interface, customers can set the parameter to be False.		
Customer Description	Setting the parameter to enable the connection of SysCoReq and SysCoAck interface to the AIU. if customer's CPU does not have the interface, and does not set False to the parameter, it is recommended to tie SysCoReq to 0.		
Engineering Description	Connect the I/O to the SysCo Engine hardware.		

TABLE 7-44: PARAMETER RELATED TO EVENTIN INTERFACE

Name: useEvenInInt			Visibility: User
	Architecture	**Release**	**Default**
	Boolean	*Boolean*	
Value	*True, False*	*True, False*	True
Constraints	Default True to ACE/CHI interface CPU, but if customer's CPU does not have the EventIn interface, customers can set the parameter to be False.		
Customer Description	Setting the parameter to enable the connection of EventInReq and EventInAck interface to the AIU. if customer's CPU does not have the interface, and does not set False to the parameter, it is recommended to tie EventInAck to EventInReq.		
Engineering Description	Connect the I/O to the SysReq Receiver hardware.		

TABLE 7-45: PARAMETER RELATED TO EVENTOUT INTERFACE

Name: useEvenOutInt			Visibility: User
	Architecture	**Release**	**Default**
	Boolean	*Boolean*	
Value	*True, False*	*True, False*	True
Constraints	Default True to ACE/CHI interface CPU, but if customer's CPU does not have the EventOut interface, customers can set the parameter to be False.		
Customer Description	Setting the parameter to enable the connection of EventOutReq and EventOutAck interface to the AIU. if customer's CPU does not have the interface, and does not set False to the parameter, it is recommended to tie EventOutReq to 0.		
Engineering Description	Connect the I/O to the SysReq Sender hardware.		

8. *Concerto User Settable Parameters*

All Concerto parameters are derived parameters.

9. *CAIU User Settable Parameters*

9.1. CAIU resource parameters

An AIU converts certain inbound native agent requests into protocol coherent transactions and allocate resources in the AIU Outstanding Transaction Table (OTT). This parameter configures the size of OTT table.

TABLE 9-1: NOTTCTRLENRIES FOR CAIU

Name: nOttCtrlEnries					Visibility: User
	Architecture		**Release**		**Default**
	Min	*Max*	*Min*	*Max*	
Value	4	128			4
Constraints	Applied to CHI-AIU and IOAIU				
Customer Description	Specify the maximum number of outstanding native transactions this AIU should support.				
Engineering Description					

TABLE 9-2: GENERIC PORTS PARAMETER FOR CAIU

Name: genericports			Visibility: User
	Architecture	Release	Default
Constraints	Applied to CHI-AIU and IOAIU		
Engineering Description	To assign user defined ports for place holder definition(Resiliency); Described in Chapter 20.1		

TABLE 9-3: MEMORY PARAMETER FOR CAIU

Name: Memory				Visibility: User
	Architecture	Release		Default
Constraints	Applied Only to IOAIU			
Engineering Description	This parameter is to assign SRAM. For the memory setting, refer Table 20-5. This is only for ACE.			

9.2. CAIU credit parameters

Specify the maximum number of CHI link credits this AIU will support. The number of credits will be specified for each flow, they define how many transactions can be in flight between an initiator and a target.

TABLE 9-4: NNATIVECREDITS PARAMETER FOR CAIU

nNativeCredits	Architecture		Release		Default
	Min	*Max*	*Min*	*Max*	
Value	2	15			2
Constraints	Applied Only to CHI-AIU				
Customer Description	Specify the maximum number of CHI link credits this AIU should support.				
Engineering Description					

Specify the maximum number of credits for coherent transactions per DCE.This should be determined based on required bandwidth and netwrok round trip latency.

TABLE 9-5: NDCECMDCREDITS FOR CAIU

nDceCmdCredits	Architecture		Release		Default
	Min	*Max*	*Min*	*Max*	
Value	2	16			2
Constraints	Applied to CHI-AIU and IOAIU				
Customer Description	Specify the maximum number of credits for coherent transactions per DCE. This should be determined based on required bandwidth and network round trip latency.				
Engineering Description					

Specify the maximum number of credits for non-coherent transactions per DMI.This should be determined based on required bandwidth and netwrok round trip latency.

TABLE 9-6: NDMICMDCREDITS FOR CAIU

nDmiCmdCredits	Architecture		Release		Default
	Min	*Max*	*Min*	*Max*	
Value	2	16			2
Constraints	Applied to CHI-AIU and IOAIU				
Customer Description	Specify the maximum number of credits for non-coherent transactions per DMI. This should be determined based on required bandwidth and network round trip latency.				
Engineering Description					

Specify the maximum number of credits for non-coherent transactions per DII.This should be determined based on required bandwidth and netwrok round trip latency.

TABLE 9-7: NDIICMDCREDITS FOR CAIU

nDiiCmdCredits	Architecture		Release		Default
	Min	*Max*	*Min*	*Max*	
Value	2	16			2
Constraints	Applied to CHI-AIU and IOAIU				
Customer Description	Specify the maximum number of credits for non-coherent transactions per DII. This should be determined based on required bandwidth and network round trip latency.				
Engineering Description					

Specify the maximum number of outstanding stash snoops this AIU should support.

TABLE 9-8: NSTASHSNPCREDITS FOR CAIU

nStashSnpCredits	Architecture		Release		Default
	Min	*Max*	*Min*	*Max*	
Value	1	8			2
Constraints	Applied Only to CHI-AIU				
Customer Description	Specify the maximum number of outstanding stash snoops this AIU should support. These are stash snoops issued on the CHI interface.				
Engineering Description	This is used for assign Ott Stash entries in CAIU. Total number of OTT entries = nOttCtrlEntries + nStshSnpCredits				

9.3. CAIU address map parameter

TABLE 9-9: FNCSRACCESS_PARAMETER

fnCsrAccess	Architecture	Release	Default
	Valid Values	*Valid values*	
Value	True, False	True, False	True
Constraints	Should be true on at-least one AIU, cannot be true for AXI AIU with NcMode as false. Applied to CHI-AIU and IOAIU		
Customer Description	Enable CSR access via this AIU		
Engineering Description	Parameter works as a reset value for CSR BAR valid bit.		

9.4. CAIU snoop filter parameters

TABLE 9-10: SNOOPFILTER_REF PARAMETER FOR CAIUS

SnoopFilter_Ref	Architecture		Release		Default
	Min	*Max*	*Min*	*Max*	
Value	0	64			0
Constraints	Applied to CHI-AIU and IOAIU.				
Customer Description	Specify the snoop filter associated with this AIU				
Engineering Description	User would need to bind CAIU into specific snoop filter, using update_object -name $caiu_name -type snoopFilter -bind $ snoop_filter_name. Archi team would want to move this parameter to DCE after NCore 3.2.				

9.5. CAIU performance counter parameters

TABLE 9-11: CAIU PERFORMANCE COUNTER PARAMETERS

nPerfCounters (CAIU/IOAIU)	Architecture		Release		Default
	Min	*Max*	*Min*	*Max*	
Value	4	16	4	8	4
Constraints	Only two valid values are supported: 4 and 8. Applied to CHI-AIU and IOAIU.				
Customer Description	Total number of performance counter in NCore Unit. Each counter can be configured to count different events present in an Ncore unit via CSRs, Please refer to the refrence manual performance counter event section.				
Engineering Description					

TABLE 9-12: CAIU LATENCY COUNTER PARAMETERS

nLatencyCounters (CAIU/IOAIU)	Architecture		Release		Default
	Min	*Max*	*Min*	*Max*	
Value	0	32	0	16	16
Constraints	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.				

9.6. CAIU processor info parameters

Ncore supports exclusive monitors by creating a basic monitor for each core in each DCE and a configurable number of tagged monitors in each DCE.

Each basic monitor implements the behavior described in the "Minimum PoS Exclusive Monitor" section in the ACE specification, and each tagged monitor implements the behavior described in the "Additional address comparison" section.

The number of cores performing an exclusive sequence **MUST** be specified per CAIU **(nProcessors)**. In the case of ACE-CAIU the ARID and AWID bits that identify the core performing an exclusive access sequence must be specified **(AxIdProcSelectBits).**

TABLE 9-13: NPROCESSOR PARAMETERS FOR CAIU

nProcessors	Architecture	Release	Default
	Enum	*Enum*	
Value	CHI-CAIU: 1, 2, 4, 8, 16, ~~32~~ IOAIU: 1, 2, 4, 8, 16, 32		1
Constraints	Applied to CHI-AIU and IOAIU.		
Customer Description	Number of Processors		
Engineering Description	CHI-AIU: SW must multipy the specified parameter by 2 before passing it on to RTL. This is to account for threads as each core can have upto two threads. For the ACE, we should not do this shift		

TABLE 9-14: AXIDPROCSELECTBITS PARAMETERS FOR CAIU

AxIdProcSelectBits	Architecture	Release	Default
	Integer Array	*Integer Array*	
Value			
Constraints	Applied Only to IOAIU.		
Customer Description	Processor Select Bits from AXID. If there is only one processor then at least one bit must be specified, instead of []		
Engineering Description			

9.7. CAIU SysCmd Hardware parameters

The following parameters are used to instantiate specific hardware within the CAIU to process sysco/event messages. The following parameters should be visible to Engining team only.

TABLE 9-15: USESYSCOENGINE PARAMETERS FOR CAIU

Name: useSysCoEngine			Visibility: Engg
	Architecture	**Release**	**Default**
	Boolean	*Boolean*	
Value	*True, False*	*True, False*	True
Constraints	Always True for ACE/CHI/AXI with Proxy Cache AIUs if useSysCoInt is True, set True to this parameter		
Customer Description			
Engineering Description	Used to instantiate SysCo Engine hardware in the AIU		

TABLE 9-16: USESYSREQSENDER PARAMETERS FOR CAIU

Name: useSysReqSender			Visibility: Engg
	Architecture	**Release**	**Default**
	Boolean	*Boolean*	
Value	*True, False*	*True, False*	True
Constraints	Always True for ACE/CHI AIUs Optional for ACE_Lite + DVM AIUs if useEventOutInt is True, set True to this parameter		
Customer Description			
Engineering Description	Used to instantiate SysReq Sender hardware in the AIU		

TABLE 9-17: USESYSREQRECEIVER PARAMETERS FOR CAIU

Name: useSysReqReceiver			Visibility: Engg
	Architecture	**Release**	**Default**
	Boolean	*Boolean*	
Value	*True, False*	*True, False*	True
Constraints	Always True for ACE/CHI AIUs if useEventInInt is True, set True to this parameter		
Customer Description			
Engineering Description	Used to instantiate SysReq Receiver hardware in the AIU		

9.8. CAIU Connectivity parameters

The following parameters are used to specify connectivity information of the CAIU. The following parameters should be visible to Engining team only.

TABLE 9-18: HEXAIUDCEVEC PARAMETERS FOR CAIU

Name: hexAiuDceVec					Visibility: Engg
	Architecture		**Release**		**Default**
	Min	*Max*	*Min*	*Max*	
Value	0	FFFFFFFF	0	FFFF	1
Constraints	Size of the vector is equal to the number of DCEs in the system. Every bit in the vector that is set to one represents a DCE at that NodeID that is connected to the AIU.				
Customer Description					
Engineering Description	This must be a port in RTL (tACHL) and tie off parameter in SW Every bit in the vector that is set to one specifies that the particular AIU is connected to the associated DCE at that NunitID				

TABLE 9-19: HEXAIUDMIVEC PARAMETERS FOR CAIU

Name: hexAiuDmiVec					Visibility: Engg
	Architecture		**Release**		**Default**
	Min	*Max*	*Min*	*Max*	
Value	0	FFFFFFFF	0	FFFF	1
Constraints	Size of the vector is equal to the number of DMIs in the system. Every bit in the vector that is set to one represents a DMI at that NodeID that is connected to the AIU.				
Customer Description					
Engineering Description	This must be a port in RTL (tACHL) and tie off parameter in SW Every bit in the vector that is set to one specifies that the particular AIU is connected to the associated DMI at that NunitID				

TABLE 9-20: HEXAIUDIIVEC PARAMETERS FOR CAIU

Name: hexAiuDiiVec					Visibility: Engg
	Architecture		**Release**		**Default**
	Min	*Max*	*Min*	*Max*	
Value	0	FFFFFFFF	0	FFFF	1
Constraints	Size of the vector is equal to the number of DIIs in the system. Every bit in the vector that is set to one represents a DII at that NodeID that is connected to the AIU.				
Customer Description					
Engineering Description	This must be a port in RTL (tACHL) and tie off parameter in SW Every bit in the vector that is set to one specifies that the particular AIU is connected to the associated DII at that NunitID				

TABLE 9-21: HEXAIUCONNECTEDDCEFUNITID PARAMETERS FOR CAIU

Name: hexAiuConnectedDceFunitId					Visibility: Engg
	Architecture		**Release**		**Default**
	Min	*Max*	*Min*	*Max*	
Value	0	FFFFFFFF	0	FFFF	1
Constraints	List of DCE Funit IDs that are connected to the AIU. This list can be ordered in Nunit ID order.				
Customer Description					
Engineering Description	This must be a port in RTL (tACHL) and tie off parameter in SW. List of DCE FuntilDs that are connected to the AIU.				

TABLE 9-22: NAIUCONNECTEDDCES PARAMETERS FOR CAIU

Name: nAiuConnectedDces					Visibility: Engg
	Architecture		**Release**		**Default**
	Min	*Max*	*Min*	*Max*	
Value	1	64	1	32	1
Constraints	Number of DCEs connected to this each AIU.				
Customer Description					
Engineering Description	Specifies the number of caching agents (AIUs) that are connected to DCE				

10. DCE User Settable Parameters

10.1. DCE resource parameters

When DCE accepts a CMDreq for processesing, it allocates an entry in the ATT (Active Transaction Table) where it stores all persistent fields from a message. The entry will remain allocated until the transaction has completed in the system. The number of entries needs to be determined by analyzing performance requirements (BW, latency) and configuring the ATT size for each DCE.

TABLE 10-1: NATTCTRLENTRIES PARAMETER

nAttCtrlEnries	Architecture		Release		Default
	Min	*Max*	*Min*	*Max*	
Value	4	64			4
Constraints					
Customer Description	Specify the maximum number of active coherent transactions tracked by each DCE.				
Engineering Description					

TABLE 10-2: NCMDSKIDBUFSIZE PARAMETER

nCMDSkidBufSize	Architecture		Release		Default
	Min	*Max*	*Min*	*Max*	
Value	4	320	4	320	16
Constraints	≥ nCMDSkidBufArb restrict granularity, supporting only sizes: nCMDSkidBufArb + {0, 4, 8, 16, 32, 64, 96, 128, 160, 192, 224, 256}				
Customer Description	Total depth of skid buffer for commands to DCE/DII and the non-coherent port of DMI. The skid buffer is used to stage transaction requests from initiator agents. The number of required entries may be determined by traffic requirements and analysis using performance modeling. This value sets the total budget of protocol credits available for distribution. CSR Address: 0xFF0				
Engineering Description	This value sets the total budget of protocol credits available for distribution to communicating initiators. It is recommended to allow at least 2 credits for each active connection.				

TABLE 10-3: NCMDSKIDBUFARB PARAMETER

nCMDSkidBufArb	Architecture		Release		Default
	Min	*Max*	*Min*	*Max*	
Value	4	64	4	64	16
Constraints	≤ nCMDSkidBufSize restricted granularity, support only sizes: 4, 8, 16, 32, 48, 64				
Customer Description	Depth of skid buffer visible to arbitration. This value determines the size of the arbitration window within which arriving requests are selected based on QoS, priority and arrival time. It is recommended to start with a reasonably value for performance analysis - the area of a skid buffer grows with the square of this number and larger options will also significantly impact timing.				
Engineering Description	This value sets the number of entries within the skid buffer that is visible to arbitration CSR Address: 0xFF0				

10.2. DCE credit parameters

TABLE 10-4: NDCERBCREDITS PARAMETER

nDceRbCredits	Architecture		Release		Default
	Min	*Max*	*Min*	*Max*	
Value	2	16			2
Constraints					
Customer Description	Specify the maximum number of DCE write request buffer credits per DMI. These credits limit number of Coherent writes and incudes snoops that can cause a write to DMI.				
Engineering Description					

TABLE 10-5: NAIUSNPCREDITS PARAMETER

nAiuSnpCredits	Architecture		Release		Default
	Min	*Max*	*Min*	*Max*	
Value	2	16			2
Constraints					
Customer Description	Specify the maximum number of snoop request credits per AIU.				
Engineering Description					

TABLE 10-6: NDMIMRDCREDITS PARAMETER

nDmiMrdCredits	Architecture		Release		Default
	Min	*Max*	*Min*	*Max*	
Value	2	16			2
Constraints					
Customer Description	Specify the maximum number of memory read credits per DMI.				
Engineering Description					

10.3. DCE snoop filter parameters

TABLE 10-7: NSETS SF CONFIGURATION PARAMETERS

nSets	Architecture		Release		Default
	Min	*Max*	*Min*	*Max*	
Value	16	1048576			16
Constraints	Number of sets must be divisible by number of DCEs in the system.				
Customer Description	Number of sets for the selected snoop filter				
Engineering Description					

TABLE 10-8: NWAYS SF CONFIGURATION PARAMETERS

nWays	Architecture		Release		Default
	Min	*Max*	*Min*	*Max*	
Value	2	32			4
Constraints					
Customer Description	Number of ways for the selected snoop filter				
Engineering Description					

TABLE 10-9: NVICTIMENTRIES SF CONFIGURATION PARAMETERS

nVictimEntries	Architecture		Release		Default
	Min	*Max*	*Min*	*Max*	
Value	0	64			2
Constraints					
Customer Description	Number of victim buffer entries for the specified snoop filter per DCE				
Engineering Description					

TABLE 10-10: APRIMARYBITS PARAMETERS

aPrimaryBits	Architecture	Release	Default
	Array of Integers	*Array of Integers*	
Value			
Constraints	Bits that select the set. They need to be as many as log2(number of sets / number of DCEs), they cannot be in the LSBs inside a cache line, and they cannot overlap with DCE interleaving bits. For example, for a system with 64B cache lines, 1024 sets and 4 DCEs interleaved on bits [11 : 10], this needs to be an 8-bit array with values that could be e.g. [15, 14, 13, 12, 9, 8, 7, 6].		
Customer Description	Set selection parameter.		
Engineering Description			

TABLE 10-11: MEMORY PARAMETER FOR DCE SNOOP FILTER

Memory	Architecture	Release	Default
Constraints			
Engineering Description	This parameter is to assign SRAM. For the memory setting, refer Table 20-5.		
NOTE: The max number of snoop filter is constrained to be no more than Snoop agents. Snoop filter must have an aiu assigned.			

10.4. DCE performance counter parameters

TABLE 10-12: DCE NPERFCOUNTERS PARAMETERS

nPerfCounters	Architecture		Release		Default
	Min	*Max*	*Min*	*Max*	
Value	4	16	4	8	4
Constraints	Only two valid values are supported. 4 and 8				
Customer Description	Total number of performance counter in Ncore Unit				
Engineering Description	Archi team would modify this as a common parameter after NCore 3.2.				

10.5. DCE exclusive monitor parameters

Ncore supports exclusive monitors (only for shareable domain) by creating a basic monitor for each core in each DCE and a configurable number of tagged monitors in each DCE. Each basic monitor implements the behavior described in the "Minimum PoS Exclusive Monitor" section in the ACE specification, and each tagged monitor implements the behavior described in the "Additional address comparison" section.

TABLE 10-13: NTAGGEDMONITORS PARAMETER

Name: nTaggedMonitors					Visibility: User
	Architecture		**Release**		**Default**
	Min	*Max*	*Min*	*Max*	
Value	0	8			0
Constraints					
Customer Description	Specify the desired number of tagged exclusive monitor per DCE instance. Note that each DCE instance will always have a basic exclusive monitor.				
Engineering Description					

10.6. DCE Connectivity parameters

The following parameters are used to specify connectivity information of the DCE. The following parameters should be visible to Engining team only.

TABLE 10-14: HEXDCECONNECTEDDMIFUNITID PARAMETER

Name: hexDceConnectedDmiFunitID					Visibility: Engg
	Architecture		**Release**		**Default**
	Min	*Max*	*Min*	*Min*	
Value	0	FFFFFFFF	0	FFFF	1
Constraint	List of DMI Funit IDs that are connected to the DCE. This is ordered in Nunit ID order , skipping DMIs not connected to the DCE.				
Customer Description					
Engineering Description	This must be a port in RTL (tACHL) and tie off parameter in SW List of DMI FuntiIDs that are connected to the DCE				

TABLE 10-15: HEXDCECONNECTEDCAFUNITID PARAMETER

Name: hexDceConnectedCaFunitID					Visibility: Engg
	Architecture		**Release**		**Default**
	Min	*Max*	*Min*	*Min*	
Value	0	FFFFFFFF	0	FFFF	1
Constraint	List of caching agent Funit IDs that are connected to the DCE. This list can be ordered in either snoop filter order or Nunit ID order				
Customer Description					
Engineering Description	This must be a port in RTL (tACHL) and tie off parameter in SW List of caching agent FuntilDs that are connected to the DCE				

TABLE 10-16: HEXDCEDMIVEC PARAMETER

Name: hexDceDmiVec					Visibility: Engg
	Architecture		**Release**		**Default**
	Min	*Max*	*Min*	*Min*	
Value	0	FFFFFFFF	0	FFFF	1
Constraint	Size of the vector is equal to the number of DMIs in the system				
Customer Description					
Engineering Description	This must be a port in RTL (tACHL) and tie off parameter in SW Every bit in the vector that is set to one specifies that the particular DCE is connected to the associated DMI at that NunitID				

TABLE 10-17: HEXDCEDMIRBOFFSET PARAMETER

Name: hexDceDmiRbOffset					Visibility: Engg
	Architecture		**Release**		**Default**
	Min	*Max*	*Min*	*Min*	
Value	0	Max 32 DMIs	0	Max 16 DMIs	1
Constraint	The max length (number of bits) is defined as number of DMIs connected to a DCE in the system multiplied by 8 Each 8-bit value represents the DMI connected to that DCE. They are ordered in the increasing order NunitID, skipping DMIs not connected to the DCE. The 8-bit offset value is calculated as follows For every DMI create a vector of all DCEs in the system. Every bit in the vector that is set to one represents a DCE at that NodeID that is connected to the DMI. For the first valid DCE in the vector the offset value is nDceRbCredits * 0 For the second valid DCE in the vector the offset value is nDceRbCredits * 1 So on and so forth This breaks down to a formula as nDceRbCredits * (Dce position - 1)				
Customer Description					
Engineering Description	This must be a port in RTL (tACHL) and tie off parameter in SW List of 8 bit values, where every 8 bit value specifies the RBID offset to be used by DCE for the DMI represented by the value. The offsets are ordered in incrementing DMI NunitID order.				

TABLE 10-18: NDCECONNECTEDCAS PARAMETER

Name: nDceConnectedCas					Visibility: Engg
	Architecture		**Release**		**Default**
	Min	*Max*	*Min*	*Min*	
Value	1	64	1	32	1
Constraint	Number of Caching agents connected to each DCE. This parameter must be same for all DCEs				
Customer Description					
Engineering Description	Specifies the number of caching agents (AIUs) that are connected to DCE				

TABLE 10-19: NDCECONNECTEDDMIS PARAMETER

Name: nDceConnectedDmis					Visibility: Engg
	Architecture		**Release**		**Default**
	Min	*Max*	*Min*	*Min*	
Value	1	32	1	16	1
Constraint	Number of DMIs connected to each DCE. This parameter must be same for all DCEs				
Customer Description					
Engineering Description	Specifies the number of DMIs that are connected to DCE				

TABLE 10-20: NDCERBCREDITS PARAMETER

Name: nDceRbCredits					Visibility: Engg
	Architecture		**Release**		**Default**
	Min	*Max*	*Min*	*Min*	
Value	2	32	2	32	2
Constraint	Number of RB credits per DCE The value is same for all DCEs and DMIs				
Customer Description					
Engineering Description	Number of RB credits per DCE				

11. DMI User Settable Parameters

11.1. DMI resource parameters

TABLE 11-1: GENERICPORTS PARAMETERS FOR DMI

genericports	Architecture	Release	Default
Constraints			
Engineering Description	To assign user defined ports for place holder definition(Resiliency); Described in Chapter 20.1		

TABLE 11-2: NRTTCTRLENTRY PARAMETERS

nRttCtrlEnries	Architecture		Release		Default
	Min	*Max*	*Min*	*Max*	
Value	4	128			4
Constraints					
Customer Description	Specify the number of allowed outstanding read transactions on the downstream AXI interface.				
Engineering Description					

TABLE 11-3: NWTTCTRLENTRY PARAMETERS

nWttCtrlEnries	Architecture		Release		Default
	Min	*Max*	*Min*	*Max*	
Value	4	64			4
Constraints					
Customer Description	Specify number of allowed outstanding write transactions on the downstream AXI interface.				
Engineering Description					

TABLE 11-4: NDMIRBCREDITS PARAMETER

nDmiRbCredits	Architecture		Release		Default
	Min	*Max*	*Min*	*Max*	
Value	2	64			2
Constraints					
Customer Description	Specify the maximum number of non-coherent write request buffer credits.				
Engineering Description					

TABLE 11-5: NCMDSKIDBUFSIZE PARAMETER

nCMDSkidBufSize	Architecture		Release		Default
	Min	Max	Min	Max	
Value	4	320	4	320	16
Constraints	≥ nCMDSkidBufArb restrict granularity, supporting only sizes: nCMDSkidBufArb + {0, 4, 8, 16, 32, 64, 96, 128, 160, 192, 224, 256}				
Customer Description	Total depth of skid buffer for commands to DCE/DII and the non-coherent port of DMI. The skid buffer is used to stage transaction requests from initiator agents. The number of required entries may be determined by traffic requirements and analysis using performance modeling. This value sets the total budget of protocol credits available for distribution.				
Engineering Description	This value sets the total budget of protocol credits available for distribution to communicating initiators. It is recommended to allow at least 2 credits for each active connection. CSR Address: 0xFF0				

TABLE 11-6: NCMDSKIDBUFARB PARAMETER

nCMDSkidBufArb	Architecture		Release		Default
	Min	*Max*	*Min*	*Max*	
Value	4	64	4	64	16
Constraints	≤ nCMDSkidBufSize restricted granularity, support only sizes: 4, 8, 16, 32, 48, 64				
Customer Description	Depth of skid buffer visible to arbitration. This value determines the size of the arbitration window within which arriving requests are selected based on QoS, priority and arrival time. It is recommended to start with a reasonably value for performance analysis - the area of a skid buffer grows with the square of this number and larger options will also significantly impact timing.				
Engineering Description	This value sets the number of entries within the skid buffer that is visible to arbitration CSR Address: 0xFF0				

TABLE 11-7: NMRDSKIDBUFSIZE PARAMETER

nMrdSkidBufSize	Architecture		Release		Default
	Min	*Max*	*Min*	*Max*	
Value	4	320	4	320	16
Constraints	≥ nMrdSkidBufSize restrict granularity, supporting only sizes: nMrdSkidBufArb + {0, 4, 8, 16, 32, 64, 96, 128, 160, 192, 224, 256}				
Customer Description	Total depth of skid buffer for coherent DMI transactions - arriving from DCE. The skid buffer is used to stage transaction requests from initiator agents. The number of required entries may be determined by traffic requirements and analysis using performance modeling. This value sets the total budget of protocol credits available for distribution.				
Engineering Description	This value sets the total budget of protocol credits available for distribution to communicating initiators. It is recommended to allow at least 2 credits for each active connection. CSR Address: 0xFE0				

TABLE 11-8: NMRDSKIDBUFARB PARAMETER

nMrdSkidBufArb	Architecture		Release		Default
	Min	*Max*	*Min*	*Max*	
Value	4	64	4	64	16
Constraints	≤ nMrdSkidBufArb restricted granularity, support only sizes: 4, 8, 16, 32, 48, 64				
Customer Description	Depth of skid buffer visible to arbitration. This value determines the size of the arbitration window within which arriving requests are selected based on QoS, priority and arrival time. It is recommended to start with a reasonably value for performance analysis - the area of a skid buffer grows with the square of this number and larger options will also significantly impact timing.				
Engineering Description	This value sets the number of entries within the skid buffer that is visible to arbitration CSR Address: 0xFE0				

TABLE 11-9: NUSEMEMRSPINTRLV PARAMETER

nUseMemRspIntrlv	Architecture		Release		Default
Value	True	False	True	False	False
Constraints					
Customer Description	Use this parameter to enable the feature of DMI can accept read data interleaving from AXI interface				
Engineering Description	To prevent deadlock issue of AXI write address channel, write response channel and read data channel, if the parameter is set to True, and read data buffer is instantiated. And DMI can accept any beat of read data of issued read request, then the read data/response channel and write response channel will never to backpressured.				

11.2. DMI address map parameters

TABLE 11-10: NADDRTRANSREGISTERS PARAMETERS

nAddrTransRegisters	Architecture		Release		Default
	Min	*Max*	*Min*	*Max*	
Value	0	4			0
Constraints					
Customer Description	Specifies the number of address translation registers that are available within DMI. These registers add capability to translate address on the AXI bus from DMI. Refer to the address translation section of the reference manual.				
Engineering Description	Need to confirm max is 4, not 8.				

11.3. DMI System Cache parameters

TABLE 11-11: DMI SYSTEM CACHE ENABLE PARAMETERS

Name: hasSysMemCache			Visibility: User
	Architecture	**Release**	**Default**
	Valid Values	*Valid values*	
Value	True, False		False
Constraints			
Customer Description	This option adds an SMC in DMI. It must be enabled when an atomic capable master is present in the system and requires at least a 4KB SMC.		
Engineering Description			

TABLE 11-12: DMI SCRATCHPAD ENABLE PARAMETERS

useScratchPad	Architecture	Release	Default
	Valid Values	*Valid values*	
Value	True, False		False
Constraints	Can be enabled only when system cache is enabled.		
Customer Description	Enable ScratchPad		
Engineering Description			

TABLE 11-13: DMI CACHE WAY PARTITIONING REGISTERS PARAMETERS

nWayPartitioningRegisters	Architecture		Release		Default
	Min	*Max*	*Min*	*Max*	
Value	0	16			0
Constraints					
Customer Description	Specifies the number of cache way partitioning registers. Each register enables configuration capability to assign specific ways to a single agent. The number of registers enabled here should be equal to maximum number of agents that will be configured for way partitioning.				
Engineering Description					

TABLE 11-14: DMI CACHE NTAGBANK CONFIGURATION PARAMETERS

nTagBanks	Architecture		Release		Default
	Min	*Max*	*Min*	*Max*	
Value	1	2			1
Constraints	Values limited to 1, 2.				
Customer Description	Number of Tag banks.				
Engineering Description					

TABLE 11-15: DMI CACHE NDATABANK CONFIGURATION PARAMETERS

nDataBanks	Architecture		Release		Default
	Min	*Max*	*Min*	*Max*	
Value	1	4			1
Constraints	Values limited to 1, 2.4				
Customer Description	Number of Data banks.				
Engineering Description					

TABLE 11-16: DMI CACHE POLICY CONFIGURATION PARAMETERS

cacheReplPolicy	Architecture	Release	Default
	Enum	*Enum*	
Value	RANDOM, NRU		RANDOM
Constraints			
Customer Description	Cache Replacement Policy		
Engineering Description			

TABLE 11-17: MEMORY PARAMETER SMC

Memory	Architecture	Release	Default
Constraints			
Engineering Description	This parameter is to assign SRAM. For the memory setting, refer Table 20-7 and Table 20-8		

11.4. DMI performance counter parameters

TABLE 11-18: DMI NPERFCOUNTERS PARAMETERS

nPerfCounters	Architecture		Release		Default
	Min	*Max*	*Min*	*Max*	
Value	4	16	4	8	4
Constraints	Ncore 3.2 only supports 4 or 8				
Customer Description	Total number of performance counter in Ncore Unit				
Engineering Description	Architechture team would modify this as a common parameter after NCore 3.2.				

TABLE 11-19: DMI LATENCY COUNTER PARAMETERS

nLatencyCounters (CAIU/IOAIU)	Architecture		Release		Default
	Min	*Max*	*Min*	*Max*	
Value	0	32	0	16	16
Constraints	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.				

11.5. DMI Atomic parameters

The SMC offers an option to include an Atomic Engine (AE). The AE supports the Far Atomic Operations (FAOs) defined in CHI-B and ACE-Lite-E interface architectures. Thus, in Ncore 3 FAOs are supported for all locations in system memory connected via the DMI

TABLE 11-20: DMI USEATOMIC PARAMETERS

Name: useAtomic			Visibitity: User
	Architecture	**Release**	**Default**
	Boolean	*Boolean*	
Value	True, False		FALSE
Constraints	Cannot be set when there is no cache.		
Customer Description	This option adds an atomic engine in DMI. It must be enabled when an atomic capable master is present in the system and requires at least a 4KB SMC.		
Engineering Description	The number of atomic engine entries is set to 4 within the DMI. Hard coded and no variable.		

11.6. DMI QoS Enhencement parameters

The customer/user of Ncore is expected to develop the following assumptions apply in this use case

1. The DMC used has 2 AXI ports one for regular traffic shown as “AXI reg” and another for high priority or real time traffic shown as “AXI high”
2. The user or customer develops “Buffer & Mux/De-Mux”block

The “Buffer & mux/de-mux” block consists of simple logic where it has a buffer that is larger than the DMC’s AXI reg port buffer. The mux/de-mux logic is responsible for routing the high priority or real time traffic to DMC’s AXI high, while all other traffic is routed to DMC’s AXI reg port. The buffer being larger than the buffer DMC’s AXI reg port buffer grantees that high priority traffic does not see head of line blocking.

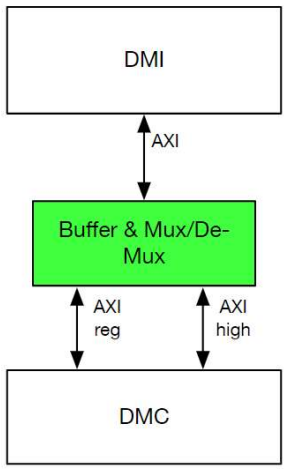


TABLE 11-21: DMIQOSTHVAL PARAMETERS

Name: DmiQoSThVal					Visibitity: User
	Architecture		**Release**		**Default**
	Min	*Max*	*min*	*max*	
Value	1	15	1	15	8
Constraints	This parameter is available only when QoS(Table 4-17) is enabled.				
Customer Description	DMI QoS threshold value. Traffic with QoS equal to or above this value are considered as high priority hard real time traffic.				
Engineering Description					

TABLE 11-22: NDMIWTTQOSRSV PARAMETERS

Name: nDmiWttQoSRsv					Visibitity: User
	Architecture		**Release**		**Default**
	Min	*Max*	*min*	*max*	
Value	1	64	1	32	1
Constraints	This parameter is available only when QoS is enabled. Maximum acceptable value must be minimum of WTT size - 1 or size of DMI non-coherent write data buffer or Coherent write data buffer. • Non-Coherent write data buffer is represented by DMI RB credits. • Coherent write data buffer is represented by number of connected DCEs multiplied by DCE RB credits per DMI. Max value = minimum of (Max WTT size - 1, DMI RB Credits - 1, DCE RB Credits - 1 * number of connected DCEs) Example: WTT size = 16 DMI RB credits = 24(non-Coherent write data buffer size) DCE RB credits = 4 and number of DCEs connected to DMI = 2 This gives the coherent write data buffer size of 4*2 = 8 As of the three numbers Coherent write data buffer size of 8 is smallest then maximum possible value is 8 - 1 = 7				
Customer Description	WTT entries in DMI reserved for high priority hard real time traffic.				
Engineering Description					

TABLE 11-23: NDMIRTTQOSRSV PARAMETERS

Name: nDmiWttQoSRsv					Visibitity: User
	Architecture		**Release**		**Default**
	Min	*Max*	*min*	*max*	
Value	1	64	1	32	1
Constraints	This parameter is available only when QoS is enabled. Maximum acceptable value must be RTT size - 1				
Customer Description	RTT entries in DMI reserved for high priority hard real time traffic.				
Engineering Description					

12. DII User Settable Parameters

12.1. DII resource parameters

TABLE 12-1: NRTTCTRLENTRY PARAMETERS

Name: nRttCtrlEnries					Visibitity: User
	Architecture		**Release**		**Default**
	Min	*Max*	*min*	*max*	
Value	4	32			4
Constraints					
Customer Description	Specify number of outstanding read transactions on the AXI interface.				
Engineering Description					

TABLE 12-2: NWTTCTRLENTRY PARAMETERS

Name: nWttCtrlEnries					Visibitity: User
	Architecture		**Release**		**Default**
	Min	*Max*	*min*	*max*	
Value	4	32			4
Constraints					
Customer Description	Specify number of outstanding write transactions on the AXI interface.				
Engineering Description					

Specify the size of the largest endpoint device connected to the DII. The size is in KBs. This size isued to achive endpoint ordering as defined by ARM CHI specification

TABLE 12-3: NLARGESTENDPOINT PARAMETER

Name: nLargestEndpoint					Visibitity: User
	Architecture		**Release**		**Default**
	Min	*Max*	*min*	*max*	
Value	4	2^39			4
Constraints					
Customer Description	Specify the size of the largest endpoint device connected to this DII. The size is in KBs. This size will be used to achieve endpoint ordering as defined by CHI architecture requirements.				
Engineering Description					

Note: Why are we allowing such a large value - could 2^{32} be sufficient?

TABLE 12-4: NDIIRBCREDITS PARAMETER

Name: nDiiRbCredits					Visibitity: User
	Architecture		**Release**		**Default**
	Min	*Max*	*min*	*max*	
Value	2	32			2
Constraints					
Customer Description	Specify the maximum number of non-coherent write request buffer credits.				
Engineering Description					

TABLE 12-5: NCMDSKIDBUFSIZE PARAMETER

nCMDSkidBufSize	**Architecture**		**Release**		**Default**
	Min	*Max*	*Min*	*Max*	
Value	4	320	4	320	16
Constraints	≥ nCMDSkidBufArb restrict granularity, supporting only sizes: nCMDSkidBufArb + {0, 4, 8, 16, 32, 64, 96, 128, 160, 192, 224, 256}				
Customer Description	Total depth of skid buffer for commands to DCE/DII and the non-coherent port of DMI. The skid buffer is used to stage transaction requests from initiator agents. The number of required entries may be determined by traffic requirements and analysis using performance modeling. This value sets the total budget of protocol credits available for distribution.				
Engineering Description	This value sets the total budget of protocol credits available for distribution to communicating initiators. It is recommended to allow at least 2 credits for each active connection. CSR Address: 0xFF0				

TABLE 12-6: NCMDSKIDBUFARB PARAMETER

nCMDSkidBufArb	**Architecture**		**Release**		**Default**
	Min	*Max*	*Min*	*Max*	
Value	4	64	4	64	16
Constraints	≤ nCMDSkidBufSize restricted granularity, support only sizes: 4, 8, 16, 32, 48, 64				
Customer Description	Depth of skid buffer visible to arbitration. This value determines the size of the arbitration window within which arriving requests are selected based on QoS, priority and arrival time. It is recommended to start with a reasonably value for performance analysis - the area of a skid buffer grows with the square of this number and larger options will also significantly impact timing.				
Engineering Description	This value sets the number of entries within the skid buffer that is visible to arbitration CSR Address: 0xFF0				

12.2. DII address map parameters

TABLE 12-7: NADDRTRANSREGISTERS PARAMETER FOR DII

Name: nLargestEndpoint					Visibitity: User
	Architecture		**Release**		**Default**
	Min	*Max*	*min*	*max*	
Value	0	4			0
Constraints					
Customer Description	Specifies the number of address translation registers that are available within the DII. Refer to the address translation capability section in the reference manual.				
Engineering Description	[XXX] From NCore's spec, the limitation is 8. Need confirmation regarding range				

12.3. DII performance monitor parameters

TABLE 12-8: DII NPERFCOUNTERS PARAMETERS

Name: nPerfCounters					Visibitity: User
	Architecture		**Release**		**Default**
	Min	*Max*	*min*	*max*	
Value	0	4			0
Constraints	Only two valid values are supported. 4 and 8				
Customer Description	Customer Description Total number of performance counter in Ncore Unit				
Engineering Description	Architechture team would modify this as a common parameter after NCore 3.2.				

TABLE 12-9: DII LATENCY COUNTER PARAMETERS

nLatencyCounters (CAIU/IOAIU)	**Architecture**		**Release**		**Default**
	Min	*Max*	*Min*	*Max*	
Value	0	32	0	16	16
Constraints	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.				

13. *DVE User Settable Parameters*

13.1. DVE resource parameters

Credit parameters and trace buffer parameters are defined at system level. Only SRAM selection will be configured in block level.

TABLE 13-1: MEMORY PARAMTER FOR DVE

Memory	**Architecture**	**Release**	**Default**
Constraints			
Engineering Description	This parameter is to assign SRAM. For the memory setting, refer Table 20-9		

13.2. DVE performance monitor parameters

TABLE 13-2: DVE NPERFCOUNTERS PARAMETERS

nPerfCounters	**Architecture**		**Release**		**Default**
	Min	*Max*	*Min*	*Max*	
Value	4	16	4	8	4
Constraints	Only two valid values are supported: 4 and 8				
Customer Description	Total number of performance counter in Ncore Unit				
Engineering Description	Archi team would modify this as a common parameter after NCore 3.2.				

13.3. System level credit parameters

NCore provides two system-level configurable parameters for DVM operation: nDvmCmdCredits and nDvmSnpCredits.

nDvmCmdCredits allows the AIU to have that many non-Sync/Sync DVMOps outstanding to DVE.

nDvmSnpCredits allows the DVE to issue that many DVMOps for snoops outstanding at a time. This value is the minimum of such outstanding DVMInv snoops supported by any AIU in the system. Note that each DVMOp snooped corresponds to 2 SNPreq messages

TABLE 13-3: NDVMCMDCREDITS PARAMETER

Name: nDvmCmdCredits			Type: int		Visibiltiy: User
	Architecture		**Release**		**Default**
	Min	*Max*	*Min*	*Max*	
Value	2	4			2
Constraints	Must be a multiple of 2.				
Customer Description	Number of DVM command credits between an AIU and a DVE.				
Engineering Description	This parameter is applicable to all AIUs that can issue DVMs.				

TABLE 13-4: NDVMSNPCREDITS PARAMETER

Name: nDvmSnpCredits		Type: Int	Visibility: User
	Architecture	**Release**	**Default**
	Enum	*Enum*	
Value	4, 6, 8		4
Constraints	Must be a multiple of 2		
Customer Description	Number of DVM snoop request credits between AIU and DVE.		
Engineering Description	This parameter is applicable to all AIUs that can accept a DVM snoop.		

14. NCAIU User Settable Parameters

14.1. NCAIU multiport parameters

TABLE 14-1: NNATIVEINTERFACEPORTS PARAMETER FOR NCAIU

nNativeInterfacePorts	Architecture		Release		Default
	Min	*Max*	*Min*	*Max*	
Value	1	8	1	8	1
Constraints	Valid values are power of 2s. 1, 2, 4, or 8.				
Customer Description	Specifies the number of native interface ports				
Engineering Description					

TABLE 14-2: ANCAIUINTVFUNC PARAMETER FOR NCAIU

aNcaiuIntvFunc	Architecture	Release	Default
Parameters	*Name*	*Name*	
	aPrimaryBits	*aPrimaryBits*	Array of integers
	aSecondaryBits	Not user visible	Array of strings
Constraints	aPrimaryBits depth depends on nNativeInterfacePorts parameter value, it will be log2 of nNativeInterfacePorts. The bits must be address bits between Max address width minus 1 and cacheline boundary address bit. For 64Bcache line it is 6. Example aPrimaryBits : [9, 8, 6] aSecondaryBits is an array of string, its depth will be same as aPrimaryBits. The string represents a hexadecimal number one hot encoded. Bits selected here cannot be same as the bits in aPrimaryBits. Example aSecondaryBits: [“'h4000”, “'h0”, “'h800”]		
Customer Description	aPrimaryBits depth depends on nNativeInterfacePorts parameter value, it will be log2 of nNativeInterfacePorts. The bits must be address bits between Max address width minus 1 and cacheline boundary address bit. For 64Bcache line it is 6. Example aPrimaryBits : [7, 6] for a 4 ports NCAIU interleaving at 64B address boundary.		
Engineering Description			

14.2. NCAIU resource parameters

TABLE 14-3: NOTTCTRLENRIES FOR NCAIU

nOttCtrlEnries	Architecture		Release		Default
	Min	*Max*	*Min*	*Max*	
Value	4	1024	4	124	32
Constraints	Must be multiple of nNativeInterfacePorts. When divided by nNativeInterfacePorts it must be less than or equal to 128. Min is for a single port				
Customer Description	Specify the maximum number of outstanding native transactions this AIU should support.				
Engineering Description					

TABLE 14-4: MEMORY PARAMETER FOR NCAIU

Memory	Architecture	Release	Default
Constraints			
Engineering Description	For the memory setting, refer Table 20-5 and Table 20-8		

14.3. NCAIU credit parameters

TABLE 14-5: NDCECMDCREDITS FOR NCAIU

nDceCmdCredits	Architecture		Release		Default
	Min	*Max*	*Min*	*Max*	
Value	2	32	2	32	32
Constraints	Must be multiple of nNativeInterfacePorts for both min and max ranges and actual value. Min is for a single port.				
Customer Description	Specify the maximum number of credits for coherent transactions per DCE. This should be determined based on required bandwidth and network round trip latency.				
Engineering Description					

TABLE 14-6: NDMICMDCREDITS FOR NCAIU

nDmiCmdCredits	Architecture		Release		Default
	Min	*Max*	*Min*	*Max*	
Value	2	32	2	32	16
Constraints	Must be multiple of nNativeInterfacePorts for both min and max ranges and actual value. Min is for a single port.				
Customer Description	Specify the maximum number of credits for non-coherent transactions per DMI. This should be determined based on required bandwidth and network round trip latency.				
Engineering Description					

TABLE 14-7: NDIICMDCREDITS FOR NCAIU

nDiiCmdCredits	Architecture		Release		Default
	Min	*Max*	*Min*	*Max*	
Value	2	32	2	32	16
Constraints	Must be multiple of nNativeInterfacePorts for both min and max ranges and actual value. Min is for a single port.				
Customer Description	Specify the maximum number of credits for non-coherent transactions per DII. This should be determined based on required bandwidth and network round trip latency.				
Engineering Description					

14.4. NCAIU address map parameter

TABLE 14-8: FNCSRACCESS_PARAMETER

fnCsrAccess	Architecture	Release	Default
	Valid Values	*Valid values*	
Value	True, False	True, False	True False for NCAIU with AXI.
Constraints	Should be true at least one AIU. Always false on coherent AXI NCAIU where nonCoherentMode (**Error! Reference source not found.**) parameter is set to false.		
Customer Description	Enables CSR access via this AIU		
Engineering Description			

14.5. NCAIU snoop filter parameters

TABLE 14-9: SNOOPFILTERASSIGNMENT PARAMETER

snoopFilterAssignment	Architecture		Release		Default
	Min	*Max*	*Min*	*Max*	
Value	0	64			0
Constraints	Only visible for (1) "AXI" with "hasProxyCache == TRUE". Also refer table in chapter 9.4.				
Customer Description	Specify the snoop filter associated with this AIU. This only applies for AIUs with proxy cache and ACE interface.				
Engineering Description	Will stay at AIU parameter at least for NCore 3.2. Already had agreed, and too late to change.				

14.6. NCAIU proxy cache parameters

Proxy cache is only supported when NCAIU native protocol is configured as “AXI”.

TABLE 14-10: HASPROXYCACHE PARAMETER

hasProxyCache	Architecture	Release	Default
	Valid Values	*Valid values*	
Value	True, False		False
Constraints	Only visible for “AXI”		
Customer Description	This option enables Proxy Cache support. This is supported only with AXI interface		
Engineering Description			

TABLE 14-11: NONCOHERENTMODE PARAMETER

Name: nonCoherentMode			Visiblity: User
	Architecture	**Release**	**Default**
	Valid Values	*Valid values*	
Value	True, False		False
Constraints	Only visible for “AXI”		
Customer Description	Only applicable if the interface is AXI. Whether the traffic on the interface shall be treated as non-coherent. If a Proxy Cache is used, the traffic is always treated as coherent. When nonCoherentMode is true, NCAIU can access CSR. When this parameter is false, NCAIU will not be able to access CSR.		
Engineering Description	For AXI when the proxy cache is not enabled then this bit specifies if the AXI going to behave as coherent or non coherent		

TABLE 14-12: PROXYCACHE NTAGBANK CONFIGURATION PARAMETERS

nTagBanks	Architecture		Release		Default
	Min	*Max*	*Min*	*Max*	
Value	1	2			1
Constraints	Values limited to 1, 2. Only visible for “AXI” with “hasProxyCache == TRUE”				
Customer Description	Number of Tag banks.				
Engineering Description					

TABLE 14-13: PROXYCACHE NDATABANK CONFIGURATION PARAMETERS

nDataBanks	Architecture		Release		Default
	Min	*Max*	*Min*	*Max*	
Value	1	4			1
Constraints	Values limited to 1, 2.4 Only visible for “AXI” with “hasProxyCache == TRUE”				
Customer Description	Number of Data banks. [MK] feedback: Constraint missing				
Engineering Description					

TABLE 14-14: PROXYCACHE POLICY CONFIGURATION PARAMETERS

cacheReplPolicy	Architecture	Release	Default
	Enum	*Enum*	
Value	RANDOM, NRU		RANDOM
Constraints	Only visible for "AXI" with "hasProxyCache == TRUE"		
Customer Description	Cache Replacement Policy		
Engineering Description			

14.7. NCAIU performance counter parameters

TABLE 14-15: NPERFCOUNTERS PARMAETER FOR NCAIU

nPerfCounters	Architecture		Release		Default
	Min	*Max*	*Min*	*Max*	
Value	4	16	4	8	4
Constraints	Only two valid values are supported. 4 and 8				
Customer Description	Total number of performance counter in Ncore Unit				
Engineering Description	Archi team would modify this as a common parameter after NCore 3.2.				

TABLE 14-16: NLATENCYCOUNTERS PARMAETER FOR NCAIU

nLatencyCounters (CAIU/IOAIU)	Architecture		Release		Default
	Min	*Max*	*Min*	*Max*	
Value	0	32	0	16	16
Constraints	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.				

14.8. NCAIU disable read data interleaving parameters

TABLE 14-17: FNDISABLERDINTERLEAVE PARMAETER FOR NCAIU

fnDisableRdInterleave	Architecture		Release		Default
	Min	*Max*	*Min*	*Max*	
Value	0	1	0	1	0
Constraints					
Customer Description	When set disables read data interleaving across different AXI IDs				
Engineering Description	When set disables read data interleaving across different AXI IDs. This parameter applies to NCAIU with AXI, ACE-Lite and ACE-Lite E ports				

14.9. NCAIU SysCmd Hardware parameters

The following parameters are used to instantiate specific hardware within the CAIU to process sysco/event messages. The following parameters should be visible to Engining team only.

TABLE 14-18: USESYSCOENGINE PARAMETERS FOR NCAIU

Name: useSysCoEngine			Visibility: Engg
	Architecture	**Release**	**Default**
	Boolean	*Boolean*	
Value	*True, False*	*True, False*	True
Constraints	Always True for ACE/CHI/AXI with Proxy Cache AIUs if useSysCoInt is True, set True to this parameter		
Customer Description			
Engineering Description	Used to instantiate SysCo Engine hardware in the AIU		

TABLE 14-19: USESYSREQSENDER PARAMETERS FOR NCAIU

Name: useSysReqSender			Visibility: Engg
	Architecture	**Release**	**Default**
	Boolean	*Boolean*	
Value	*True, False*	*True, False*	True
Constraints	Always True for ACE/CHI AIUs Optional for ACE_Lite + DVM AIUs if useEventOutInt is True, set True to this parameter		
Customer Description			
Engineering Description	Used to instantiate SysReq Sender hardware in the AIU		

TABLE 14-20: USESYSREQRECEIVER PARAMETERS FOR NCAIU

Name: useSysReqReceiver			Visibility: Engg
	Architecture	**Release**	**Default**
	Boolean	*Boolean*	
Value	*True, False*	*True, False*	True
Constraints	Always True for ACE/CHI AIUs if useEventInInt is True, set True to this parameter		
Customer Description			
Engineering Description	Used to instantiate SysReq Receiver hardware in the AIU		

14.10. NCAIU Connectivity parameters

The following parameters are used to specify connectivity information of the NCAIU. The following parameters should be visible to Engining team only.

TABLE 14-21: HEXAIUDCEVEC PARAMETERS FOR NCAIU

Name: hexAiuDceVec					Visibility: Engg
	Architecture		**Release**		**Default**
	Min	*Max*	*Min*	*Max*	
Value	0	FFFFFFFF	0	FFFF	1
Constraints	Size of the vector is equal to the number of DCEs in the system. Every bit in the vector that is set to one represents a DCE at that NodeID that is connected to the AIU.				
Customer Description					
Engineering Description	This must be a port in RTL (tACHL) and tie off parameter in SW Every bit in the vector that is set to one specifies that the particular AIU is connected to the associated DCE at that NunitID				

TABLE 14-22: HEXAIUDMIVEC PARAMETERS FOR NCAIU

Name: hexAiuDmiVec					Visibility: Engg
	Architecture		**Release**		**Default**
	Min	*Max*	*Min*	*Max*	
Value	0	FFFFFFFF	0	FFFF	1
Constraints	Size of the vector is equal to the number of DMIs in the system. Every bit in the vector that is set to one represents a DMI at that NodeID that is connected to the AIU.				
Customer Description					
Engineering Description	This must be a port in RTL (tACHL) and tie off parameter in SW Every bit in the vector that is set to one specifies that the particular AIU is connected to the associated DMI at that NunitID				

TABLE 14-23: HEXAIUDIIVEC PARAMETERS FOR NCAIU

Name: hexAiuDiiVec					Visibility: Engg
	Architecture		**Release**		**Default**
	Min	*Max*	*Min*	*Max*	
Value	0	FFFFFFFF	0	FFFF	1
Constraints	Size of the vector is equal to the number of DIIs in the system. Every bit in the vector that is set to one represents a DII at that NodeID that is connected to the AIU.				
Customer Description					
Engineering Description	This must be a port in RTL (tACHL) and tie off parameter in SW Every bit in the vector that is set to one specifies that the particular AIU is connected to the associated DII at that NunitID				

TABLE 14-24: HEXAIUCONNECTEDDCEFUNITID PARAMETERS FOR NCAIU

Name: hexAiuConnectedDceFunitId					Visibility: Engg
	Architecture		**Release**		**Default**
	Min	*Max*	*Min*	*Max*	
Value	0	FFFFFFFF	0	FFFF	1
Constraints	List of DCE Funit IDs that are connected to the AIU. This list can be ordered in Nunit ID order.				
Customer Description					
Engineering Description	This must be a port in RTL (tACHL) and tie off parameter in SW. List of DCE FuntilDs that are connected to the AIU.				

TABLE 14-25: NAIUCONNECTEDDCES PARAMETERS FOR NCAIU

Name: nAiuConnectedDces					Visibility: Engg
	Architecture		**Release**		**Default**
	Min	*Max*	*Min*	*Max*	
Value	1	64	1	32	1
Constraints	Number of DCEs connected to this each AIU.				
Customer Description					
Engineering Description	Specifies the number of caching agents (AIUs) that are connected to DCE				

15. CCP User Settable Parameters

The CCP is a configurable Cache IP block. It is commonly used for all the IPs which requires Cache access. Currently it is being used by Proxy Cache in IO-AIU, and SMC in DMI

TABLE 15-1: NSETS PARAMETERS OF CCP

Name: nSets			Visibitity: User
	Architecture	**Release**	**Default**
	Valid Values	*Boolean*	
Value	16', '32', '64', '128', '256', '512', '1024', '2048', '4096', '8192		16
Constraints	The number of sets per data bank must be greater than the number of data banks. The number of sets per tag bank must be greater than the number of tag banks. Expect log2(nSets) bits for primary selection bits. Must be multiple of nNativeInterfacePorts for both min and max ranges and actual value.		
Customer Description	Specify the number of sets/entries in the Cache.		
Engineering Description			

TABLE 15-2: NWAYS PARAMETER OF CCP

Name: nWays					Visibitity: User
	Architecture		**Release**		**Default**
	Min	*Max*	*min*	*max*	
Value	2	16			2
Constraints					
Customer Description	Specify the number of sets/entries in the Cache.				
Engineering Description					

TABLE 15-3: USESCRATCHPAD PARAMETER OF CCP

Name: useScratchPad			Visibitity: User
	Architecture	**Release**	**Default**
	Boolean	*Boolean*	
Value	True, False		FALSE
Constraints			
Customer Description	Enable Scratchpad. The visibility will be overridden based on block type.		
Engineering Description			

TABLE 15-4: PRISUBDIAGADDRBITS PARAMETERS

Name: PriSubDiagAddrBits			Visibitity: User
	Architecture	**Release**	**Default**
	Array of strings	*Array of strings*	
Value			
Constraints	Prisubdiagaddrbits depth must be log2 (nSets/nNativeInterfacePorts). The bits must be address bits between Max address width minus 1 and cacheline boundary address bit. For 64Bcache line it is 6.They cannot include address bits used in aPrimaryBits of aNcaiuIntvFunc and address bits used in aPrimaryAiuPortBits		
Customer Description	Specify address bits to be used as primary set select bits.		
Engineering Description			

TABLE 15-5: TAGBANKSELBITS PARAMETERS

Name: TagBankSelBits			Visibitity: User
	Architecture	**Release**	**Default**
	Array of strings	*Array of strings*	
Value			
Constraints			
Customer Description	The tag bank select bit values must be unique. The tag bank selecion bit must be one of the primary set selection bits. The number of tag bank bits must be log2(nTagBanks).		
Engineering Description			

TABLE 15-6: DATABANKSELBITS PARAMETERS

Name: DataBankSelBits			Visibitity: User
	Architecture	**Release**	**Default**
	Array of strings	*Array of strings*	
Value			
Constraints	The data bank select bit values must be unique. The data bank bits must be one of the primary set selection bits. The number of data bank bits must be log2(nDataBanks) bits.		
Customer Description	Specify data bank select bit. This bit must be one of the bits from the primary select bits.		
Engineering Description			

16. *Legato User Settable Parameters*

Async adapter and dw_adapter are automatically inserted. Async adpaters are inserted between different clock domains, and dw_adpaters are inserted if there is mismatch between input and output of the link.
Data width inside of the network would be configured using portDataWidth of the sym_switch and sym_buf_switch. Network parameter is not being supported at NCore 3.2
Some of the derived/fixed parameters have been described in this section (because many of the engineers are reading only user settable part) but they may be moved to a separate "derived/fixed" chapter in a later version of the specification

16.1. sym_switch/sym_buf_switch

The sym_buf_switch supports configurable buffers at the ingress port of the switches.

TABLE 16-1: SYM_BUF_SWITCH AND SYM_SWITCH PARAMETER: PORTDATAWIDTH

Name: portDataWidth			Visibility: User
	Architecture	**Release**	**Default**
	Valid values	*Valid Values*	
Value	64, 128, 256		
Constraints			
Customer Description	Data width of all ports of the switch		
Engineering Description	Applied to sym_switch and sym_buf_switch		

TABLE 16-2: SYM_BUF_SWITCH PARAMETER: INPUTBUFFERDEPTH

Name: inputBufferDepth			Visibility: User
	Architecture	**Release**	**Default**
Value	[0, 2, 4, 8, 12, 16, 24, 32]		2
Constraints			
Customer Description	Buffer depth is buffer depth at **input port** (Layer 0) If we configure inputBufferDepth 0, **sym_switch** is configured.		
Engineering Description	NCore 3.2 supports only Buffer Layer 0 buffers.		

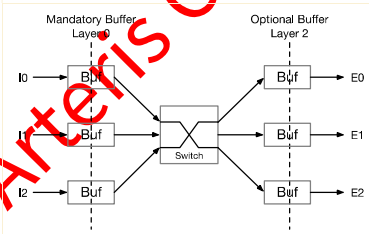


FIGURE 16-1: SYM_BUF_SWITCH IN CDTI, WITH ONLY ONE VC

16.2. sym_async_adapter

Clock adapers require the specification of two different FIFO depths:

- The depth of the synchronizers used for signals that cross domains for metastability reasons.
- The depth of the circular FIFO used to transfer data from one side to the other and the depth affects functional throughput.

The synchronizer depth is configurable to supporting a circular FIFO (added from NCore 3.2)

- A new system parameter called **syncDepth** is added to configure synchronizer depth of sym_async_adapter. This new parameter will be used to set the depth of the synchronizers.
- Circular FIFO depth = Math.ceil(2*(syncDepth+1.5)).
- NCore 3.2 supports syncDepth values of 2, 3, and 4 only

16.3. chi_async_adapter

sym_async_adapter is for SMI interface, and chi_async_adapter is to support CHI interface. It has a slave CHI interface and a master CHI interface, each interface has its own clock. Depth of the circular FIFO are calculated according to the number credit if the CHI interface. No user settable parameters.

16.4. sym_rate_adapter

Ncore 3.2 does not support rate adapters

16.5. dw_adapter

No user settable parameter. Buffer depth is calculated inside of the block

If **pipeforward** and **pipeBackward** are set true, the depth parameters **dfDepth** and **hfDepth** must be set to at least 2, otherwise bubbles will be inserted into the data stream.

16.6. sym_pipe_adapter

TABLE 16-3: SYM_PIPE_ADAPTER PARAMETER: DEPTH

Name: depth			Visibility: User
	Architecture	**Release**	**Default**
Value	[0,1,2,3]	[1,2]	2
Constraints			
Customer Description	Fifo depth inside of the sym_pipe_adapter		
Engineering Description	Depth 1 is expected for CSR network – mostly the network which doesn't require performance.		

17. *Derived/Fixed Socket Parameters*

17.1. AXI_Interface

TABLE 17-1: AXI_INTERFACE FIXED PARAMETERS

Parameter Name	Type		Default	Min	Max	Enum	Description
wResp	Integer	Fixed	2	2	4		
wWUser	Integer	Fixed	0	0	16	Not being used	
wBUser	Integer	Fixed	0	0	200		
wRUser	Integer	Fixed	0	0	200	Not being used	
wLen	Integer	Fixed	8	8	8		
wSize	Integer	Fixed	3	3	4	Only 3. Because we are not supporting 512.	
wLock	Integer	Fixed	1	0	1	Always 1	
wQos	Integer	Fixed	0	0	4	['0', '4']	
wRegion	Integer	Fixed	0	0	4	Fixed as 0	
wProt	Integer	Fixed	3			Fixed as 3	

17.2. APB_Interface

TABLE 17-2: APB_INTERFACE FIXED PARAMETERS

Parameter Name	Type		Default	Min	Max	Enum	Description
wAddr	Integer	Fixed	12	12	64		
wData	Integer	Fixed	32	8	64	['8', '16', '32', '64']	
wProt	Integer	Fixed	0	0	3	['0', '3']	
wStrb	Integer	Fixed	0	0	4	['0', '1', '2', '4']	
wPSlverr	Integer	Fixed	0	0	1		

17.3. ACE_Interface

TABLE 17-3: ACE_INTERFACE USER SETTABLE PARAMETERS

Parameter Name	Type		Default	Min	Max	Description/Derivation
eUnique	Integer	Eng. Param.	1	0	1	
wCdData	Integer	Fixed	0			
wSnoop	Integer	Eng. Param.	3	3	3	
eAc	Integer	Fixed	1	1	1	
wResp	Integer	Fixed	4	2	4	
eDomain	Integer	Fixed	1	1	1	
useQos	Boolean	Derived				useQoS: system parameter
wQos	Integer	Derived				wQos = (useQos) ? 4 : 0;

17.4. ACELITE_E_Interface

TABLE 17-4: ACELITEE_INTERFACE DERIVED PARAMETERS

Parameter Name	Type		Default	Min	Max	Description/Derivation
wLoop	Integer	Fixed	0	0	0	
eTrace	Integer	Fixed	1	1	1	MAES-3605, changed from 0 to 1 to support Trace signal at NCore 3.2.
eUnique	Integer	Fixed	0	0	0	
wCdData	Integer	Fixed	0			
wSnoop	Integer	Derived	3	3	4	wSnoop = eStash == 1 ? 4 : 3;
eStash	Integer	Fixed	1	1	1	
eAtomic	Integer	Fixed	1	1	1	
eDomain	Integer	Fixed	1	1	1	

17.5. ACELITE_Interface

TABLE 17-5: ACELITE_INTERFACE DERIVED PARAMETERS

Parameter Name	Type		Default	Min	Max	Description/Derivation
wLoop	wLoop	Fixed	0	0	0	
eTrace	eTrace	Fixed	0	0	0	
eUnique	eUnique	Fixed	0	0	0	
wCdData	wCdData	Fixed	0			
wSnoop	wSnoop	Fixed	3	3	3	
eStash	eStash	Fixed	0	0	0	
eAtomic	eAtomic	Fixed	0	0	0	
eDomain	eDomain	Fixed	1	1	1	

17.6. CHI_A_Interface

TABLE 17-6: CHI_A_INTERFACE DERIVED PARAMETERS

Parameter Name	Type		Default	Min/ Max	Description/Derivation
SrcID	Integer	Derived	7	7	SrcID = NodeID_Width;
TgtID	Integer	Derived	7	7	TgtID = NodeID_Width;
TxnID	Integer	Fixed	8	8	
REQ_Opcode	Integer	Fixed	5	5	
RSP_Opcode	Integer	Fixed	4	4	
SNP_Opcode	Integer	Fixed	4	4	
DAT_Opcode	Integer	Fixed	3	3	
Size	Integer	Fixed	3	3	
wAddr	Integer	Fixed	44	44	Width of physical address
NS	Integer	Fixed	1	1	
LikelyShared	Integer	Fixed	1	1	
AllowRetry	Integer	Fixed	1	1	
Order	Integer	Fixed	2	2	
PCrdType	Integer	Fixed	2	2	
MemAttr	Integer	Fixed	4	4	
SnpAttr	Integer	Fixed	2	2	
LPID	Integer	Fixed	3	3	
Excl	Integer	Fixed	1	1	
ExCompAck	Integer	Fixed	1	1	
TraceTag	Integer	Fixed	1	1	
DAT_RSVDC	Integer	Fixed	0	16	Permitted RSVDC bus widths RSVDC = {0, 4, 8, 12, 16}
RespErr	Integer	Fixed	2	2	
Resp	Integer	Fixed	3	3	
FwdState	Integer	Fixed	3	3	
DBID	Integer	Fixed	8	8	
CCID	Integer	Fixed	2	2	
DataID	Integer	Fixed	2	2	
BE	Integer	Derived	8	8/64	BE = wData/8 wData = {64, 128, 256, 512}
wQos	Integer	Fixed	4	4	
wReqflit	Integer	Derived	65	94	wReqflit = wQos + TgtID + SrcID + TxnID + Opcode + Size + wAddr + NS + LikelyShared + AllowRetry + Order + PCrdType + MemAttr + SnpAttr + LPID + Excl + ExCompAck + REQ_RSVDC;
wRspflit	Integer	Derived	45	45	wRspflit = wQos + TgtID + SrcID + TxnID + Opcode + RespErr + Resp + DBID + PCrdType;
wSnpflit	Integer	Derived	65	65	wSnpflit = wQos + SrcID + TxnID + Opcode + wAddr + NS - 3;

TABLE 17-7: CHI_A_INTERFACE DERIVED PARAMETERS

Parameter Name	Type		Default	Min/ Max	Description/Derivation
wDatflit	Integer	Derived	190	190	wDatflit = wQos + TgtID + SrcID + TxnID + Opcode + RespErr + Resp + DBID + CCID + DataID + DAT_RSVDC + BE + wData;

17.7. CHI_B_Interface

TABLE 17-8: CHI_B_INTERFACE DERIVED PARAMETERS

Parameter Name	Type		Default	Min/ Max	Description/Derivation
SrcID	Integer	Derived	7	7/11	SrcID = NodeID_Width;
TgtID	Integer	Derived	7	7/11	TgtID = NodeID_Width;
TxnID	Integer	Fixed	8	8	
ReturnNID	Integer	Derived	7	7/11	ReturnNID = NodeID_Width;
StashNIDValid	Integer	Fixed	1	1	
ReturnTxnID	Integer	Fixed	8	8	
REQ_Opcode	Integer	Fixed	6	6	
RSP_Opcode	Integer	Fixed	4	4	
SNP_Opcode	Integer	Fixed	5	5	
DAT_Opcode	Integer	Fixed	3	3	
Size	Integer	Fixed	3	3	
wAddr	Integer	Fixed	44	44/52	Physical address width wAddr = {44, 48, 52}
NS	Integer	Fixed	1	1	
LikelyShared	Integer	Fixed	1	1	
AllowRetry	Integer	Fixed	1	1	
Order	Integer	Fixed	2	2	
PCrdType	Integer	Fixed	4	4	
MemAttr	Integer	Fixed	4	4	
SnpAttr	Integer	Fixed	1	1	
LPID	Integer	Fixed	5	5	
Excl	Integer	Fixed	1	1	
ExCompAck	Integer	Fixed	1	1	
TraceTag	Integer	Fixed	1	1	
DAT_RSVDC	Integer	Fixed	0	0/32	Permitted RSVDC bus widths RSVDC = {0, 4, 8, 12, 16, 24, 32}
RespErr	Integer	Fixed	2	2	
Resp	Integer	Fixed	3	3	
FwdState	Integer	Fixed	3	3	
DBID	Integer	Fixed	8	8	
FwdNID	Integer	Derived	1	1	FwdNID = NodeID_Width;
FwdTxnID	Integer	Fixed	8	8	
DoNotGoToSD	Integer	Fixed	1	1	

TABLE 17-9: CHI_B_INTERFACE DERIVED PARAMETERS

Parameter Name	Type		Default	Min/ Max	Description/Derivation
RetToSrc	Integer	Fixed	1	1	
Homenode_ID	Integer	Derived	7	7	Homenode_ID = NodeID_Width;
CCID	Integer	Fixed	2	2	
DataID	Integer	Fixed	2	2	
BE	Integer	Derived	8	64	BE = wData/8; wData = { 64, 128, 256}
wQos	Integer	Fixed	4	4	
wPoison	Integer	Derived	2	4	wPoison = enPoison ? (wData/64) : 0;
wReqflit	Integer	Derived	95	95	wReqflit = wQos + TgtID + SrcID + TxnID + ReturnNID + StashNIDValid + ReturnTxnID + Opcode + Size + wAddr + NS + LikelyShared + AllowRetry + Order + PCrdType + MemAttr + SnpAttr + LPID + Excl + ExpCompAck + TraceTag + REQ_RSVDC;
wRspflit	Integer	Derived	34	34	wRspflit = wQos + TgtID + SrcID + TxnID + Opcode + RespErr + Resp + FwdState + DBID + PCrdType + TraceTag;
wDatflit	Integer	Derived	125	125	wDatflit = wQos + TgtID + SrcID + TxnID + Homenode_ID + Opcode + RespErr + Resp + FwdState + DBID + CCID + DataID + TraceTag + DAT_RSVDC + BE + wPoison + wData;
wSnpflit	Integer	Derived	70	70	wSnpflit = wQos + SrcID + TxnID + FwdNID + FwdTxnID + Opcode + wAddr + NS + DoNotGoToSD + RetToSrc + TraceTag - 3;

18. *Derived/Fixed Concerto Parameters*

18.1. ConcertoC SMI Param

TABLE 18-1: CONCERTOCSMIPARAM PARAMETERS

Parameter Name	Type		Default	Min/ Max	Description/Derivation
wTargetId	Integer	Derived			wTargetId = wFUnitId + wFPortId;
wInitiatorId	Integer	Derived			wInitiatorId = wFUnitId + wFPortId;
wMsgId	Integer	Derived			wMsgId = wMessageId;
wAddr	Integer	Derived	0		Derived by mapper code max. of wAddr of all the sockets
wMPF1	Integer	Derived			wMPF1 = max({1+wFUnitId, 1+wMaxChiNodeId, wArgV, wAXIFldSet, wTargetId, wVMIDExt, wFlowId, wInitiatorId, wMsgId});
wMPF2	Integer	Derived			wMPF2 = max({(1+wLPId), (1+wFlowId), wDvmSnpUnqId, wMsgId});
wMPF3	Integer	Derived			wMPF3 = max({wFUnitId, wDvmSnpPartId, wFlowId});
wDId	Integer	Derived			wDId = wFUnitId
nBEPerDW	Integer	Fixed	8		
wBEPerDW	Integer	Fixed	8		

wProtPerDW	Integer	Derived	0	0/8	wProtPerDW = 0;
					if (ResilienceEnable)
					{if TIResiliencyProtectionType == SECDED) {
					wProtPerDW = 8; }
					if (TIResiliencyProtectionType == PARITY) {
					wProtPerDW = 1;
					}
					}

wAuxPerDW	Integer	Fixed	0	0/32	
wDPPerBeat	Integer				Possibly not being used
wDataBitsPerDW	Integer	Fixed	64	64	
wDBadPerDW	Integer	Fixed	1	1	
wDPPerDW	Integer	Derived			wDPPerDW = wDataBitsPerDW + wBEPerDW + wDBadPerDW + wDWId + wProtPerDW + wAuxPerDW;
nSmiVC	Integer	Fixed	1	1	
wSmiTid	Integer	Derived			wSmiTid = wTargetId;
wSmiSid	Integer	Derived			wSmiSid = wInitaitorId;
wSmiType	Integer	Derived			wSmiType = wCMType;
wSmiMsgId	Integer	Derived			wSmiMsgId = wMsgId;
wSmiUser	Integer	Derived			wSmiUser = wHProt;

TABLE 18-2: CONCERTOCSMIPARAM PARAMETERS

Parameter Name	Type		Default	Min/ Max	Description/Derivation
wSmiSteer	Integer	Derived			wSmiSteer = wSteering;
wSmiTier	Integer	Derived			wSmiTier = wTTier;
wSmiQos	Integer	Derived			wSmiQos = wQL;
wSmiPri	Integer	Derived			wSmiPri = wPriority;
wSmiNDPLen	Integer	Fixed	8	8	
wSmiNDP	Integer	Will be derived			This will be defined at port level.
wSmiErr	Integer	Fixed	1	1	
wSmiRoute	Integer	Fixed	0	0	
wSmiClass	Integer	Fixed	0	0	
wSmiSeqnum	Integer	Fixed	0	0	
wSmiAddr	Integer	Fixed	0	0	
wSmiLen	Integer	Fixed	0	0	
wSmiVNid	Integer	Fixed	0	0	
wSmiProt	Integer	Fixed	0	0	
wSmiTxnHdr	Integer	Fixed	0	0	
nSmiDPvc	Integer	Fixed	1	1	
wSmiDPlast	Integer	Fixed	1	1	
wSmiDPdata	Integer	Derived		128 or 256	Will be defined at block level wSmiDPdata: ncore3 uses 256 max. 512 is not verified
wSmiDPuser	Integer	Fixed	0	0	
wSmiDPbe	Integer	Fixed	0	0	
wSmiDPid	Integer	Fixed	0	0	
wSmiDPerr	Integer	Fixed	0	0	
wSmiDPresp	Integer	Fixed	0	0	
wSmiDPdummy	Integer	Fixed	0	0	

18.2. ConcertoC Param

TABLE 18-3: CONCERTOCPARAM PARAMETERS

Parameter Name	Type	Origin	Default	Min/ Max	Description/Derivation
wCacheLine	Integer	Fixed			Cache line width in byte 64B Cacye Line
wDWId	Integer	Fixed			Number of Bits Identifing a DW Within a CG
wDBad	Integer	Fixed	1		Width of the signal Dbad in bits. When set, it indicates that a data DW is corrupted (i.e. Bad) and therefore must not be consumed in a computation
wSysReqOp	Integer	Fixed	4		
wValid	Integer	Fixed	1		
wReady	Integer	Fixed	1		
wLast	Integer	Fixed	1		
wStashFUnitId	Integer				wStashFUnitId = wFUnitId;
wStashNId	Integer				wStashNId = wStashFUnitId;
HProtEnable	Boolean	Fixed	False		HProt is being defined…?
TTierEnable	Boolean	Fixed	False		Not used in Ncore 3.2
QLEnable	Boolean	Fixed	False		Not used in Ncore 3.2
SteeringEnable	Boolean	Fixed	False		Not used in Ncore 3.2
PriorityEnable	Boolean	Fixed	True		
wTargetId	Integer	Derived			wTargetId = wFUnitId + wFPortId; (from Common)
wInitiatorId	Integer	Derived			wInitiatorId = wFUnitId + wFPortId; (from Common)
wCMType	Integer	Fixed	8	8/8	
wMessageId	Integer	Derived			wMessageId = max({log2MaxAiuCredits, log2MaxDceCredits, log2MaxDmiCredits, log2MaxDiiCredits});

wHProt	Integer	Derived	0	0/12	if (! ResilienceEnable) { wHProt = Integer(0); }
					else {
					if (TIResilienceProtectionType == NONE) {
					wHProt = 0;
					} else
					if (TIResilienceProtectionType == PARITY) {
					wHProt = 1;
					} else {
					auto temp = wTargetId + wInitiatorId
					+ wCMType + wMessageId;
					int64_t ecc_width = 3;
					while ($2^{(ecc\_width - 1)}$ < (temp + ecc_width)) {
					ecc_width += 1;
					}
					wHProt = ecc_width;
					}
					}

wTTier	Integer	Fixed	0	0/4	
wSteering	Integer	Fixed	0	0/4	

TABLE 18-4: CONCERTOCPARAM PARAMETERS

Parameter Name	Type	Origin	Default	Min/ Max	Description/Derivation
wPriority	Integer	Derived		0/4	wPriority = useQos ? 3 : 0;
wQL	Integer	Fixed	0	0/4	
wCMHeader	Integer	Derived			wCMHeader = wTargetId + wInitiatorId + wCMType + wMessageId + wHProt + wTTier + wSteering + wPriority + wQL;
wCMStatus	Integer	Fixed	8	8	
wVZ	Integer	Fixed	1	1	
wCA	Integer	Fixed	1	1	
wAC	Integer	Fixed	1	1	
wCH	Integer	Fixed	1	1	
wST	Integer	Fixed	1	1	
wEN	Integer	Fixed	1	1	
wES	Integer	Fixed	1	1	
wNS	Integer	Fixed	1	1	
wPR	Integer	Fixed	1	1	
wOR	Integer	Fixed	2	2	
wLK	Integer	Fixed	2	2	
wRL	Integer	Fixed	2	2	
wTM	Integer	Fixed	1	1	
wUP	Integer	Fixed	2	2	
wPrimary	Integer	Fixed	1	1	
wMW	Integer	Fixed	1	1	
wEO	Integer	Fixed	0		
wSize	Integer	Fixed	3	3/4	
wIntfSize	Integer	Fixed	2	2/3	
wTOF	Integer	Fixed	3	1/3	
wQoS	Integer	Derived	4	0 or 4	wQos = useQos ? 4 : 0;
wTNType	Integer	Fixed	8	8	
wAddr	Integer	Derived			From NC_ConcertoCSMIParams.json
wMPF1	Integer	Derived		8/12	From NC_ConcertoCSMIParams.json
wMPF2	Integer	Derived		6/12	From NC_ConcertoCSMIParams.json
wMPF3	Integer	Derived		5/12	From NC_ConcertoCSMIParams.json
wBId	Integer	Derived			From NC_ConcertoCSMIParams.json
wRBID	Integer	Derived			From NC_ConcertoCSMIParams.json ??
wRType	Integer	Fixed	1	1	
wNdpAux	Integer	Derived		0/16	Derivation is in mapping code = max {ArUser, AwUser}

TABLE 18-5: CONCERTOCPARAM PARAMETERS

Parameter Name	Type	Origin	Default	Min/ Max	Description/Derivation
wNdpProt	Integer				Is it being used?
wRMessageId	Integer			0/12	wRMessageId = wMessageId;
wTNMsg	Integer			0/16	
ECMType	Integer				Is it being used? If there is no default vaule, Maestro is set it as 0
wArgV	Integer		6	3/8	MAES-3383. Default is changed from 3 to 6.
wFlowId	Integer	Derived	5	5/10	Derivation is in mapping code: max {ArId, AwId}
wLPId	Integer		0	0/5	Derivation is in mapping code • ACE: Determined as log2 (number of processors in the largest cluster) • CHI_A: 3 (deprecated) • CHI_B: 5

18.3. ConcertoC RequestMessageFields

TABLE 18-6: CONCERTOCREQUESTMESSAGEFIELDS PARAMETERS

Parameter Name	Type	Description/Derivation
wCMDNdp	Integer	wCMDNdp = wCMStatus + wVZ+wCA + wAC + wCH + wST + wEN + wES + wNS + wPR + wOR + wLK + wRL + wTM + wSize + wIntfSize + wTOF + wQoS + wAddr + wMPF1 + wMPF2 + wDId+ wNdpAux + wCMDMProt;
wSYSNdp	Integer	wSYSNdp = wCMStatus + wSysReqOp + wRMessageId + wTM + wSYSMProt;
wSNPNdp	Integer	wSNPNdp = wCMStatus + wVZ + wCA + wAC + wNS + wPR + wRL + wTM+wUP+wIntfSize + wTOF+ wQoS+ wAddr+ wMPF1 + wMPF2 + wMPF3+ wDId+ wRBID+ wNdpAux+ wSNPMProt;
wMRDNdp	Integer	wMRDNdp = wCMStatus + wAC + wNS+ wPR+ wRL+ wTM + wSize + wIntfSize+ wQoS+ wAddr+ wMPF1 + wMPF2 + wNdpAux + wMRDMProt;
wUPDNdp	Integer	wUPDNdp = wCMStatus + wNS + wAddr + wUPDMProt + wQos + wTM;
wHNTNdp	Integer	this transaction type will not implemented in Ncore 3.2
wSTRNdp	Integer	wSTRNdp = wCMStatus+ wMPF1 + wMPF2 + wRBID + wRMessageId + wIntfSize + wTM + wSTRMProt;
wTUNNdp	Integer	
wRBRNdp	Integer	wRBRNdp = wCMStatus + wVZ + wCA + wAC + wNS + wPR+ wRL+ wMW + wSize+ wTOF+ wQoS + wAddr + wMPF1+ wRType+ wRBID + wRBRMProt+ wNdpAux + wTM;
wRBUNdp	Integer	wRBUNdp = wCMStatus + wRL + wRBID + wTM + wRBUMProt
wDTRNdp	Integer	wDTRNdp = wCMStatus + wRL + wTM + wMPF1 + wNdpAux + wRMessageId + wDTRMProt;
wDTWNdp	Integer	wDTWNdp = wCMStatus+ wRL+ wTM + wPrimary + wMPF1 + wMPF2 + wRBID+ wNdpAux + wDTWMProt + wIntfSize;
wDTWDBGNdp	Integer	wDTWDBGNdp = wCMSTatus + wRT + wTM + wNdpAux + wDTWDBGMProt

wCMDMProt	Integer	if (! ResilienceEnable) {wCMDMProt = 0; }
		else {
		if (TIResiliencyProtectionType == NONE)
		{wCMDMProt = 0;
		} else if (TIResiliencyProtectionType == PARITY)
		{wCMDMProt = 1;
		} else {
		auto temp = wCMStatus+ wVZ + wCA + wAC+ wCH"+ wST + wEN
		+ wES+ wNS + wPR + wOR + wLK + wRL + wTM + wSize
		+ wIntfSize + wTOF + wQoS + wAddr + wMPF1 + wMPF2
		+ wDId + wNdpAux;
		int64_t ecc_width = 3;
		while (std::pow(2, ecc_width - 1) < (temp + ecc_width)) {
		ecc_width += 1;
		}
		wCMDMProt = ecc_width;
		}
		}

TABLE 18-7: CONCERTOCREQUESTMESSAGEFIELDS PARAMETERS

Parameter Name	Type	Description/Derivation

wSYSMProt	Integer	if (! ResilienceEnable) {wSYSMProt = 0; }
		else {
		if (TIResiliencyProtectionType == NONE) {
		wSYSMProt = 0;
		} else if (TIResiliencyProtectionType == PARITY) {
		wSYSMProt = 1;
		} else {
		auto temp = wCMStatus + wSysReqOp + wRMessageId + wTM;
		int64_t ecc_width = 3;
		while (std::pow(2, ecc_width - 1) < (temp + ecc_width)) {
		ecc_width += 1;
		}
		wSYSMProt = ecc_width;
		}
		}

wSNPMProt	Integer	if (! ResilienceEnable) {wSNPMProt = 0;}
		else {
		if (TIResiliencyProtectionType == NONE) {
		wSNPMProt = 0;
		} else if (TIResiliencyProtectionType == PARITY) {
		wSNPMProt = 1;
		} else {
		auto temp = wCMStatus + wVZ + wCA + wAC + wNS + wPR + wRL
		+ wTM + wUP + wIntfSize + wTOF + wQoS + wAddr
		+ wMPF1+ wMPF2 + wMPF3 + wDId+ wRBID + wNdpAux;
		int64_t ecc_width = 3;
		while (std::pow(2, ecc_width - 1) < (temp + ecc_width)) {
		ecc_width += 1;
		}
		wSNPMProt = ecc_width;
		}

wMRDMProt	Integer	if (! ResilienceEnable) {wMRDMProt = 0; }
		else if (TIResiliencyProtectionType == NONE) {
		wMRDMProt = 0;
		} else if (TIResiliencyProtectionType == PARITY) {
		wMRDMProt = 1;
		} else {
		auto temp = wCMStatus+ wAC+ wNS+ wPR+ wRL+ wTM + wSize
		+ wIntfSize+ wQoS+ wAddr+ wMPF1+ wMPF2 + wNdpAux
		int64_t ecc_width = 3;
		while (std::pow(2, ecc_width - 1) < (temp + ecc_width)) {
		ecc_width += 1;
		}
		wMRDMProt = ecc_width;
		}
		}

wHNTMProt	Integer	
wTUNMProt	Integer	

TABLE 18-8: CONCERTOCREQUESTMESSAGEFIELDS PARAMETERS

Parameter Name	Type	Description/Derivation

wUPDMProt	Integer	if (! ResilienceEnable) {wUPDMProt = 0; }
		else {if (TIResiliencyProtectionType == NONE) {
		wUPDMProt = 0;
		} else if (TIResiliencyProtectionType == PARITY) {
		wUPDMProt = 1;
		} else {
		auto temp = wCMStatus + wNS+ wAddr+ wQos + wTM;
		int64_t ecc_width = 3;
		while (std::pow(2, ecc_width - 1) < (temp + ecc_width)) {
		ecc_width += 1;
		}
		wUPDMProt = ecc_width;
		}
		}

wSTRMProt	Integer	if (! ResilienceEnable) {wSTRMProt = 0;}
		else {if (TIResiliencyProtectionType == NONE) {
		wSTRMProt = 0;
		} else if (TIResiliencyProtectionType == PARITY) {
		wSTRMProt = 1;
		} else {
		auto temp = wCMStatus + wMPF1 + wMPF2 + wRBID + wRMessageId
		+ wIntfSize + wTM;
		int64_t ecc_width = 3;
		while (std::pow(2, ecc_width - 1) < (temp + ecc_width)) {
		ecc_width += 1;
		}
		wSTRMProt = ecc_width;
		}
		}

wRBRMProt	Integer	if (! ResilienceEnable) {wRBRMProt = 0; }
		else { if (TIResiliencyProtectionType == NONE) {
		wRBRMProt = 0;
		} else if (TIResiliencyProtectionType == PARITY) {
		wRBRMProt = 1;
		} else {
		auto temp = wCMStatus + wVZ + wCA + wAC + wNS + wPR + wRL
		+ wMW + wSize+ wTOF + wQoS + wAddr + wMPF1
		+ wRType + wRBID + wNdpAux + wTM;
		int64_t ecc_width = 3;
		while (std::pow(2, ecc_width - 1) < (temp + ecc_width)) {
		ecc_width += 1;
		}
		wRBRMProt = ecc_width;
		}
		}

TABLE 18-9: CONCERTOCREQUESTMESSAGEFIELDS PARAMETERS

Parameter Name	Type	Description/Derivation

wRBUMProt	Integer	if (! ResilienceEnable) {wRBUMProt = 0; }
		else {if (TIResiliencyProtectionType == NONE) {
		wRBUMProt = 0;
		} else if (TIResiliencyProtectionType == PARITY) {
		wRBUMProt = 1;
		} else {
		auto temp = wCMStatus + wRL + wRBID + wTM;
		int64_t ecc_width = 3;
		while (std::pow(2, ecc_width - 1) <)temp + ecc_width)) {
		ecc_width += 1;
		}
		wRBUMProt = ecc_width;
		}
		}

wDTRMProt	Integer	if (! ResilienceEnable) { wDTRMProt = 0; }
		else {if (TIResiliencyProtectionType == NONE) {
		wDTRMProt = 0;
		} else if (TIResiliencyProtectionType == PARITY) {
		wDTRMProt = 1;
		} else {
		auto temp = wCMStatus + wRL + wTM + wMPF1 + wNdpAux
		+ wRMessageId;
		int64_t ecc_width = 3;
		while (std::pow(2, ecc_width - 1) < (temp + ecc_width)) {
		ecc_width += 1;
		}
		wDTRMProt = ecc_width;
		}
		}

wDTWMProt	Integer	if (! ResilienceEnable) {wDTWMProt = 0; }
		else {if (TIResiliencyProtectionType == NONE) {
		wDTWMProt = 0;
		} else if (TIResiliencyProtectionType == PARITY) {
		wDTWMProt = 1;
		} else {
		auto temp = wCMStatus + wRL + wTM + wPrimary + wMPF1 + wMPF2
		+ wRBID + wNdpAux + wIntfSize ;
		int64_t ecc_width = 3;
		while (std::pow(2, ecc_width - 1) < (temp + ecc_width)) {
		ecc_width += 1;
		}
		wDTWMProt = ecc_width;
		}
		}

TABLE 18-10: CONCERTOCREQUESTMESSAGEFIELDS PARAMETERS

Parameter Name	Type	Description/Derivation

wDTWDBGMProt	Integer	if (! ResilienceEnable) {wDTWDBGMProt = 0; }
		else {if (TIResiliencyProtectionType == NONE) {
		wDTWDBGMProt = 0;
		} else if (TIResiliencyProtectionType == PARITY) {
		wDTWDBGMProt = 1;
		} else {
		auto temp = wCMStatus + wRL + wTM + wPrimary + wMPF1+ wMPF
		+ wRBID + wNdpAux+ wIntfSize;
		int64_t ecc_width = 3;
		while (std::pow(2, ecc_width - 1) < (temp + ecc_width)) {
		ecc_width += 1;
		}
		wDTWDBGMProt = ecc_width;
		}
		}

18.4. ConcertoCResponseMessageFields

TABLE 18-11: CONCERTOCRESPONSEMESSAGEFIELDS PARAMETERS

Parameter Name	Type	Description/Derivation
wCCMDrspNdp	Integer	wCCMDrspNdp = wCMStatus + wRMessageId + wTM + wCCMDrspMProt
wSYSrspNdp	Integer	wSYSrspNdp = wCMStatus + wRMessageId + wTM + wSYSrspMProt
wNCCMDrspNdp	Integer	wNCCMDrspNdp = wCMStatus+ wRMessageId + wTM+ wNCCMDrspMProt
wSNPrspNdp	Integer	wSNPrspNdp = wCMStatus + wIntfSize + wMPF1 + wRMessageId + wTM + wSNPrspMProt
wDTWrspNdp	Integer	wDTWrspNdp = wCMStatus + wRMessageId + wRL + wTM + wDTWrspMProt
wDTWDBGrspNdp	Integer	wDTWDBGrspNdp = wCMStatus + wRMessageId + wRL + wTM + wDTWDBGrspMProt
wDTRrspNdp	Integer	wDTRrspNdp = wCMStatus + wRMessageId + wTM + wDTRrspMProt
wHNTrspNdp	Integer	
wMRDrspNdp	Integer	wMRDrspNdp = wCMStatus + wRMessageId + wTM + wMRDrspMProt
wSTRrspNdp	Integer	wSTRrspNdp = wCMStatus + wRMessageId + wTM + wSTRrspMProt;
wUPDrspNdp	Integer	wUPDrspNdp = wCMStatus+ wRMessageId+ wTM + wUPDrspMProt
wRBRrspNdp	Integer	wRBRrspNdp = wCMStatus + wRMessageId +wTM + wRBRrspMProt
wRBUrspNdp	Integer	wRBUrspNdp = wCMStatus + wRMessageId + wTM+ wRBUrspMProt
wCMPrspNdp	Integer	wCMPrspNdp = wCMStatus+ wRMessageId + wTM + wCMPrspMProt
wCMErspNdp	Integer	
wTUNrspNdp	Integer	
wTRErspNdp	Integer	

wCCMDrspMProt	Integer	if (! ResilienceEnable) { wCCMDrspMProt = 0; }
		else {
		if (TIResiliencyProtectionType == NONE) {
		wCCMDrspMProt = 0;
		} else if (TIResiliencyProtectionType == PARITY) {
		wCCMDrspMProt = 1;
		} else {
		auto temp = wCMStatus + wRMessageId + wTM;
		int64_t ecc_width = 3;
		while (std::pow(2, ecc_width - 1) < (temp + ecc_width)) {
		ecc_width += 1;
		}
		wCCMDrspMProt = ecc_width;
		}
		}

wSYSrspMProt	Integer	if (! ResilienceEnable) {wSYSrspMProt = 0; }
		else {
		if (TIResiliencyProtectionType == NONE) {
		wSYSrspMProt = 0;
		} else if (TIResiliencyProtectionType == PARITY) {
		wSYSrspMProt = 1;
		} else {
		auto temp = wCMStatus + wRMessageId +wTM
		int64_t ecc_width = 3;
		while (std::pow(2, ecc_width - 1) < (temp + ecc_width)) {
		ecc_width += 1;
		}
		wSYSrspMProt = ecc_width
		}
		}

TABLE 18-12: CONCERTOCRESPONSEMESSAGEFIELDS PARAMETERS

Parameter Name	Type	Description/Derivation

wNCCMDrspMProt	Integer	if (! ResilienceEnable) {wNCCMDrspMProt = 0; }
		else {
		if (TIResiliencyProtectionType == NONE) {
		wNCCMDrspMProt = 0;
		} else if (TIResiliencyProtectionType == PARITY) {
		wNCCMDrspMProt = 1;
		} else {
		auto temp = wCMStatus + wRMessageId + wTM;
		int64_t ecc_width = 3;
		while (std::pow(2, ecc_width - 1) < (temp + ecc_width)) {
		ecc_width += 1;
		}
		wNCCMDrspMProt = ecc_width;
		}
		}

wSNPrspMProt	Integer	if (! ResilienceEnable) {wSNPrspMProt = 0; }
		else {
		if (TIResiliencyProtectionType == NONE) {
		wSNPrspMProt = 0;
		} else if (TIResiliencyProtectionType == PARITY) {
		wSNPrspMProt = 1;
		} else {
		auto temp = wCMStatus + wInfSize + wMPF1 + wRMessageId +wTM;
		int64_t ecc_width = 3;
		while (std::pow(2, ecc_width - 1) < (temp + ecc_width)) {
		ecc_width += 1;"
		}
		wSNPrspMProt = ecc_width;
		}
		}

wDTWrspMProt	Integer	if (! ResilienceEnable) {wDTWrspMProt = 0; }
		else {
		if (TIResiliencyProtectionType == NONE) {
		wDTWrspMProt = 0;
		} else if (TIResiliencyProtectionType == PARITY) {
		wDTWrspMProt = 1;
		} else {
		auto temp = wCMStatus + wRMessageId + wRL + wTM;
		int64_t ecc_width = 3;
		while (std::pow(2, ecc_width - 1) < (temp + ecc_width)) {
		ecc_width += 1;",
		};
		wDTWrspMProt = ecc_width;
		}
		}

TABLE 18-13: CONCERTOCRESPONSEMESSAGEFIELDS PARAMETERS

Parameter Name	Type	Description/Derivation

wDTWDBGrspMProt	Integer	if (! ResilienceEnable) {wDTWDBGrspMProt = 0; }
		else { if (TIResiliencyProtectionType == NONE) {
		wDTWDBGrspMProt = 0;
		} else if (TIResiliencyProtectionType == PARITY) {
		wDTWDBGrspMProt = 1;
		} else {
		auto temp = wCMStatus + wRMessageId + wRL + wTM;
		int64_t ecc_width = 3;
		while (std::pow(2, ecc_width - 1) < temp + ecc_width) {
		ecc_width += 1;
		}
		wDTWDBGrspMProt = ecc_width;
		}
		}

wDTRrspMProt	Integer	if (! ResilienceEnable) {wDTRrspMProt = 0; }
		else {if (TIResiliencyProtectionType == NONE) {
		wDTRrspMProt = 0;
		} else if (TIResiliencyProtectionType == PARITY) {
		wDTRrspMProt = 1;
		} else {
		auto temp = wCMStatus + wRMessageId + wTM;
		int64_t ecc_width = 3;
		while (std::pow(2, ecc_width - 1) < (temp + ecc_width)) {
		ecc_width += 1;
		}
		wDTRrspMProt = ecc_width;
		}
		}

wHNTrspMProt	Integer	

wMRDrspMProt	Integer	if (! ResilienceEnable) {wMRDrspMProt = 0; }
		else {if (TIResiliencyProtectionType == NONE) {
		wMRDrspMProt = 0;
		} else if (TIResiliencyProtectionType == PARITY) {
		wMRDrspMProt = 1;
		} else {
		auto temp = wCMStatus + wRMessageId + wTM;
		int64_t ecc_width = 3;
		while (std::pow(2, ecc_width - 1) < (temp + ecc_width)) {
		ecc_width += 1;
		}
		wMRDrspMProt = ecc_width;
		}
		}

wSTRrspMProt	Integer	if (! ResilienceEnable) {wSTRrspMProt = 0; }
		else {if (TIResiliencyProtectionType == NONE) {
		wSTRrspMProt = 0;
		} else if (TIResiliencyProtectionType == PARITY) {
		wSTRrspMProt = 1;
		} else {
		auto temp = wCMStatus + wRMessageId +wTM;
		int64_t ecc_width = 3;
		while (std::pow(2, ecc_width - 1) < (temp + ecc_width)) {
		ecc_width += 1;
		}
		wSTRrspMProt = ecc_width;
		}
		}

TABLE 18-14: CONCERTOCRESPONSEMESSAGEFIELDS PARAMETERS

Parameter Name	Type	Description/Derivation

wUPDrspMProt	Integer	if (! ResilienceEnable) {wUPDrspMProt = 0; }
		else {if (TIResiliencyProtectionType == NONE) {
		wUPDrspMProt = 0;
		} else if (TIResiliencyProtectionType == PARITY) {
		wUPDrspMProt = 1;
		} else {
		auto temp = wCMStatus + wRMessageId + wTM;
		int64_t ecc_width = 3;
		while (std::pow(2, ecc_width - 1) < (temp + ecc_width)) {
		ecc_width += 1;
		}
		wUPDrspMProt = ecc_width;
		}
		}

wRBRrspMProt	Integer	if (! ResilienceEnable) {wRBRrspMProt = 0; }
		else {if (TIResiliencyProtectionType == NONE) {
		wRBRrspMProt = 0;
		} else if (TIResiliencyProtectionType == PARITY) {
		wRBRrspMProt = 1;
		} else {
		auto temp = wCMStatus + wRMessageId +wTM;
		int64_t ecc_width = 3;
		while (std::pow(2, ecc_width - 1) < (temp + ecc_width)) {
		ecc_width += 1;
		}
		wRBRrspMProt = ecc_width;
		}
		}

wRBUrspMProt	Integer	if (! ResilienceEnable) {wRBUrspMProt = 0; }
		else {if (TIResiliencyProtectionType == NONE) {
		wRBUrspMProt = 0;
		} else if (TIResiliencyProtectionType == PARITY) {
		wRBUrspMProt = 1;
		} else {
		auto temp = wCMStatus + wRMessageId +wTM;
		int64_t ecc_width = 3;
		while (std::pow(2, ecc_width - 1) < temp + ecc_width) {
		ecc_width += 1;
		}
		wRBUrspMProt = ecc_width;
		}
		}

wCMPrspMProt	Integer	if (! ResilienceEnable) {wCMPrspMProt = 0; }
		else {if (TIResiliencyProtectionType == NONE)
		{wCMPrspMProt = 0;
		} else if (TIResiliencyProtectionType == PARITY)
		{wCMPrspMProt = 1;
		} else {
		auto temp = wCMStatus + wRMessageId + wTM;
		int64_t ecc_width = 3;
		while (std::pow(2, ecc_width - 1) < temp + ecc_width) {
		ecc_width += 1;
		}
		wCMPrspMProt = ecc_width;
		}
		}

19. *Legato Derived/Fixed Parameters*

19.1. PMA

A Power Management Adapter (PMA) will be instantiated, when power domains support dynamic control through a P-channel

- If power domain is configured as dynamic (can be turned off by user), then a PMA will be allocated for all the clock domains inside of the power domain.
- If a power domain is configured as always on (will not be turned off in any case), then a PMA will be allocate when the clock domain can be turned off by a user signal (clock: external)
- PMA components do not have a CSR interface.

NOTE: PMA doesn't have any user settable/derived parameter for NCore 3.2.

FIGURE 19-1: PMA IN CLOCK DOMAIN

19.2. Sym_async_adapter

TABLE 19-1: SYM_ASYNC_ADAPTER

Parameter Name	Default	Ranges	NCore 3.2	Comments
Async	false	True/False	Derived	
Depth			Derived	Circular FIFO depth of the sym_async_adpater would be derived by this system configuration value • syncDepth: 2 --> circular fifo depth of sym_async_adapter: 8 • syncDepth: 3 --> circular fifo depth of sym_async_adapter: 10 • syncDepth: 4 --> circular fifo depth of sym_async_adapter: 12
interfaces. inPmaControlInterface			Fixed	If IN clock interface is switchable this interface should exist. Otherwise, _SKIP_ = true.
interfaces. outPmaControlInterface			Fixed	If OUT clock interface is switchable this interface should exist. Otherwise, _SKIP_ = true.
interfaces. inProtectionInterface	_SKIP_ =True		Fixed	
Interfaces. outProtectionInterface	_SKIP =True		Fixed	

Depth:

- Depth parameter will be initially defined by Network parameter, and user will have override option.

Async:

- When the two clocks into sym_async adapter are from different clock domains, then async is set to true.
- When they are from different clock sub domains and the same clock domain, then async is set to false.
- User will not be allowed to override this parameter.

19.3. Sym_buf_switch

All parameters for this element are not GUI visible

TABLE 19-2: SYM_BUF_SWITCH

Parameter Name	Default	Ranges	NCore 3.2	Comments
arbType. egress	arb_rr1	arb_rr1, arb_pri_rr1, arb_fifo	Fixed	
bufLayer0. circular	false	True/False	**Derived**	Circular will be true when depth of the buffer is greater than 2.
bufLayer0. pipeForward	True	True/False	Fixed	If bufLayer1 and bufLayer2 have 0 depth, buflayer0 pipeForward must be true. (from CPR)
bufLayer2. circular	False	True/False	Fixed	
bufLayer2. depth	0	Power of two: Min:0 Max:32	Fixed	
bufLayer2. pipeBackward	True	True/False	Fixed	This will be fixed at NCor 3.2 but description added to let the readers know the default value
bufLayer2. pipeForward	True	True	Fixed	This will be fixed at NCor 3.2 but description added to let the readers know the default value
interfaces. protectionInterface			SKIP_ = True (at R1)	
numPri	1		derived	

Circular parameter derivation:

Circular will be true when depth of the buffer is greater than 2

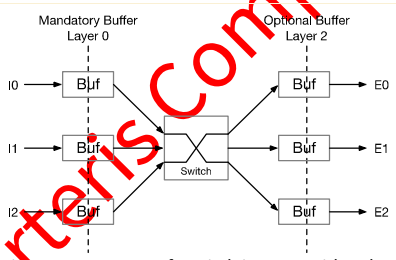


Figure 19-2: Sym_Buf_Switch in CDTI, with only one VC

19.3.1. Configuration details

Default configuration:

- bufLayer0.pipeForward = True
- bufLayer0.depth = 2
- bufLayer2.pipebackward = True
- bufLayer2.pipeForward =True
- bufLayer2.depth = 0

Circular parameter:

- Circular default value from Network: False

Circular = true/false does not affect function or performance, but timing and power. When circular is false, the output stage of the FIFO is always the same register, so it has better output timing. However, it has worse power, because when the FIFO is READ, all the registers with data get clocked as the data shifts forward. When circular is true, the FIFO uses read and write pointers, so only one register is being written or read at a time. It can have better power, because only the pointers and one register at most would clock in one cycle, but the output timing is worse, because there is a mux selecting which register to read for the output of the FIFO.

19.4. Sym_ibuf_switch

NOTE: NCore 3.2 does not support sym_ibuf_switch.

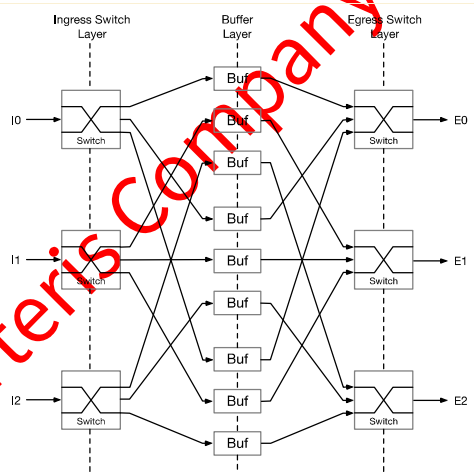


FIGURE 19-3: SYM_BUF_SWITCH IN CDTI

TABLE 19-3: SYM_IBUF_SWITCH

Parameter Name	Default	Ranges	NCore 3.2	GUI-Visibility
arbType.egress	arb_rr1	arb_rr1, arb_pri_rr1, arb_fifo	Fixed	No
Circular	False	True/False	Derived	No
numPri	1		Derived	No

19.5. Width/Rate_adapter

WidthAdapters

Ncore 3.x architecture supports different widths of networks between agents (64, 128, 256 bits)
Connections between receivers and transmitters with different widths require a WidthAdapter.
A WidthAdapter converts a sequence of phits belonging to a packet arriving from a narrow interface to the wide interface.

This avoids using only part of the wide output interface's bandwidth, which would propagate downstream. A WidthAdapter will assemble a wider phit by storing:

- at least one phit entry of the width of the outgoing port
- one entry with the difference in width between the input and the output port

A WidthAdapter will introduce additional bubbles into the down stream traffic.
A WidthAdapter converting from wide interface to narrow interface may use a single wide entry to hold a phit while breaking it down into a stream of consecutive, narrow phits.
A WidthAdapter shall track up to 4 transactions and detect the boundary between packets having a different TxnID

TABLE 19-4: NINPUTWIDTH PARAMETERS FOR WIDTHADAPTER

Name: nInputWidth					Visibility: Engg
	Architecture		**Release**		**Default**
	Min	*Max*	*Min*	*Max*	
Value	64	256	64	256	
Constraints	nInputWidth != nOutputWidth				
Customer Description					
Engineering Description	This value is a derived parameter based on the width of the source feeding this block. Maestro derives this parameter from the {FUnit, Switch}.sender.width connected to the input of the WidthAdapter.				

TABLE 19-5: NOUTPUTWIDTH PARAMETERS FOR WIDTHADAPTER

Name: nOutputWidth					Visibility: Engg
	Architecture		**Release**		**Default**
	Min	*Max*	*Min*	*Max*	
Value	64	256	64	256	
Constraints	nInputWidth != nOutputWidth				
Customer Description					
Engineering Description	This value is a derived parameter based on the width of the source feeding this block. Maestro derives this parameter from the {FUnit, Switch}.receiver.width connected to the input of the WidthAdapter.				

TABLE 19-6: BOOLPIPELINE PARAMETERS FOR WIDTHADAPTER

Name: boolPipeline					Visibility: User
	Architecture		*Release*		*Default*
Value	*True*	*False*	*True*	*False*	*False*
Constraints	nDepth ≤ 1				
Customer Description	Force insertion of at least one pipeline stage for timing reasons				
Engineering Description	Setting this parameter to True will override nDepth == 0 and force the insertion of at least one pipeline satage into the WidthAdapter. The stting has no effect if nDepth > 0.				

TABLE 19-7: NDEPTH PARAMETERS FOR WIDTHADAPTER

Name: nDepth					Visibility: Engg
	Architecture		*Release*		*Default*
	Min	*Max*	*Min*	*Max*	
Value	*0*	*4 x nTxnSize / nOutputWidth*	*0*	*4 x nTxnSize / nOutputWidth*	
Constraints	nDepth ≤ 1				
Customer Description	Add additional buffer stages to the WidthAdapter - this makes it a combined Width-Rate-Adapter				
Engineering Description	Add additional buffer stages to the WidthAdpater so that backpressure into the adapter will not immediately stall upstream, bubbles created by the width conversion will be squashed. The supported max. amount of buffer inserted will be 4 full transactions				
Note: TxnSize = 64 bytes = 512 bits					

RateAdapters

Rate adapters will explicitly be instantiated by the user.

A RateAdapter will be used when a packet, consiting of multiple phits, may contain bubbles.
The rate adapter's function is, to aggregate temporally separated pieces/phits of a transaction, and retransmit them as consecutive sequence to a downstream receiver.
The Legato interconnect does not support transmission of flits belonging to different transactions.
Rate adapters may be used to level out fluctuations in input rate, even when the arrival rate ≥ departure rate for a short time, at the cost of increased buffering

- Rate adapters always have the same width on the input and the output port
- A rate adapter implements a FiFo-Queue where the first phit of a packet (flit) will not signal valid to the downstream receiver until the entire packet has been assembled in the queue.
- A rate adapter has to implement sufficient storage to hold at least one full packet - n buffer entries organized as width bits
- number of entries n = txn_size/port_width

Pipeline support shall be supported (improved timing), adding one additional storage entry of width bits to receive the first phit for the next transaction.
Additional entries may be specified if the designer desires to optimize bursty traffic in front of a congested switch.

This will support more than a single transaction to be forwarded in an uninterrupted burst.
A rate adapter will introduce additional latency of m cycles:

- m ≥ number of phits per transaction + 1

A width adapter shall track up to 4 transactions and detect the boundary between packets having a different TxnID

TABLE 19-8: NWIDTH PARAMETERS FOR RATEADAPTER

Name: nWidth					Visibility: Engg
	Architecture		*Release*		*Default*
	Min	*Max*	*Min*	*Max*	
Value	*64*	*256*	*64*	*256*	
Constraints	nWidth == nInputWidth == nOutputWidth				
Customer Description					
Engineering Description	The width value is a derived parameter based on the width of both, the source feeding this block and the destination of the output. Maestro derives this parameter from the {FUnit, Switch}.sender.width connected to the input of the RateAdapter				

TABLE 19-9: NDEPTH PARAMETERS FOR WIDTHADAPTER

Name: nDepth					Visibility: User
	Architecture		*Release*		*Default*
	Min	*Max*	*Min*	*Max*	
Value	*0*	*4 x nTxnSize / nOutputWidth*	*0*	*4 x nTxnSize / nOutputWidth*	
Constraints	nDepth ≤ 1				
Customer Description	Defines the depth of the RateAdapter				
Engineering Description	Add additional buffer stages to the connection so that backpressure will not immediately stall upstream, bubbles in the stream will be squashed. The supported max. amount of buffer inserted will be 4 full transactions, the min. amount of buffer space will be 1 full transaction				
Note: TxnSize = 64 bytes = 512 bits					

Software (Maestro) Support

Maestro shall support automated insertion of width adapters:
When source and destination of a network segment have different width
The decision shall be made based on:

- nInputWidth = {switch, FUnit} transmitter.width
- nOutputWidth = {switch, FUnit} receiver.width

The automatically generated WidthAdapter shall be customer configurable by changing the default settings of the following parameters:

- nDepth (default = 0) to configure additional buffer stages
- boolPipelined (default = false)

Maestro shall support user configurable insertion and removal of RateAdapters

- UI shall provide a means to select a network connection between two FUnits or an FUnit and a switch

The manually inserted RateAdapter shall be configurable by UI

- nDepth (default = TxnSize) to configure buffer stages
- nDepth shall be derived from the network segment where the user chose to insert the adapter
- When a user attempts to insert a rate adapter on a segment connecting a WidthAdapter output to a receiver, Maestro shall offer to parametrize the widthAdapter to increase depth instead (do we need a forced override to insert a RateAdapter?)
- When a user attempts to insert a rate adapter in front of a WidthAdapter - Maestro shall issue a warning, this is useless and only adds latency, recommend to parametrize the width adapter instead
- Future versions of the RateAdapter may support different different clock domains for input and output ports

TABLE 19-10: INSERTION RULES

	Input < Output	Input = Output	Input > Output	Description
Type	Width Adapter	Rate Adapter	Width/Rate Adapter	Adapter type depends on the interface configuration
Rule	Automatic Insertion	Insertion by User	Automatic Insertion	When input and output do not have the same width, a Width Adapter will be required
Configurability	Automatic Insertion = Yes nInputWidth nOutputWidth	Automatic Insertion = No	Automatic Insertion = Yes nInputWidth nOutputWidth	
	User boolPipeline nDepth[1]	User Insertion nWidth nDepth boolPipeline	User boolPipeline nDepth[1]	
nDepth	Automatic 1xnInputWidth + 1xnOutputWidth User +nxnOutputWidth	User Parameter based on rate difference ≥nxnWidth	Automatic ≥1xnInputWidth User +nxnOutputWidth	Automatic insertion will always use the minimum size required for the functionality User may configure additional storage in Maestro's UI

Notes: 1. Optional, additional buffer stages for rate adaptation

19.6. Sym_pipe_adpater

NOTE: sym_pipe_adapter will not have user settable parameter. pipeBackward/pipeForward will be always true, and fifo_depth will use default value.

TABLE 19-11: SYM_PIPE_ADAPTER

Parameter Name	Default	Ranges	NCore 3.2	GUI-Visibility	Comments
Circular	true	True/False	Fixed	No	
Depth	2	**Power of two:** **Min:** - 0 **Max:** - 1K*8/wData	Fixed	No	
Split	false	True/False	Fixed	No	
interfaces. protectionInterface			_SKIP_ =True (at R1)	No	

19.7. Interrupt

Interrupts will not be aggregated within Ncore - the user needs to wire them outside of Ncore.

19.8. Parameter for CSR network

In the CSR network only atui_apb, atut_apb, and sym_switch/sym_buf_switch/ will be used.

After timing analysis, the user must insert sym_pipe_adapter manually.

No parameter will be visible to user.

No parameter is user settable. The next chapter is only for referring fixed values.

19.9. chi_async_adapter

sym_async_adapter is for SMI interface, and chi_async_adapter is to support CHI interface. It has a slave CHI interface and a master CHI interface, each interface has its own clock.

The circular FIFO depth of the chi_async_adpater is calculated according to the number of Chi request credit. (reqcredits = nCHIReqInFlight + 1.)

19.10. CSR fixed parameters

This chapter describes fixed parameters for CSR network

19.10.1. Atut_apb parameters

TABLE 19-12: ATUT_APB BLOCK PARAMETERS

Parameter	Default	Ranges	CSR network	Description
apbSlvLut			Derived	
apbSlvLut.addr	default		Derived	
apbSlvLut.chipSel			Derived	
canReceiveNarrows	true	True/False	Derived	
ctlPipeCtxt	0	0 .. 9	0	
ctlPipeReq	0	0,1,2	0	
ctlPipeResp	0	0,1,2	0	
enBufWrite	false	True/False	False	
enPathLookup	false		Fixed true	When there is a tree structure in the CSR network, no route field is needed in the packet. Whether this is needed or not will be a function of the CSR network topology (would be required for a mesh depending on the routes used, even if there was only one initiator.)
exclusivesSupported	false		False	
fixedSupported	false		Fixed false	
idCompMask	[' true']		[] It must be fixed to an empty entry.	Not really applicable because APB doesn't have ID.
incrSupported	false		Fixed true	
Interfaces				
interfaces.apbInterface				
interfaces.apbRegInterface			No APB register interface	
interfaces.atpReqInterface			Derived	
interfaces.atpRespInterface			Derived	
interfaces.clkInterface			Derived	
interfaces.intInterface			Derived	
interfaces.pmaControlInterface			Derived	
interfaces.statsInterface			Derived	
mapBaseAddr	user		Derived	
mapBaseMask	user		Derived	
maxOutRd	1		Fixed as 1	
maxOutTotal	1		Fixed as 1	
maxOutWr	1		Fixed as 1	

Parameter	Default	Ranges	CSR network	Description
nExclEntries	4	2^N with N = 0 .. 3	Fixed 0	0 means no exclusive monitor.
narrowSupported	false		Fixed false	
nodeId	0		Derived	
numPri	1		Derived	
pathLut			Derived	
pathLut.route			Derived	
pathLut.targ_id			Derived	
pipeLevelApb	0	0,1,2	Fixed 2	For timing reasons this should be 1 or 2. This is a block level interface
pipeLevelAtp	0	0,1,2	Fixed 2	For timing reasons this should be 1 or 2. This is a block level interface
pipeLevelLut	0	0,1,2	Fixed 2	If a pathLut existed, should be 1 or 2.
pipeLevelSmi	2	0,1,2	Fixed 0	This could be 0. Internal interface
readInterleaveSupported	true		Fixed False	
rdEn	true	True/False	Always true	
smiDpknumPri	1		Derived	This needs to be the numPri for the CSR network , which should be 1
smiPktnumPri	1		Derived	This needs to be the numPri for the CSR network , which should be 1
timeoutErrChk	false		False	
timeoutErrCount	512		0	
timeoutUseExternalValue	0		Fixed 1	
wApbSlvDec	2		Derived	
wDataMax	64		Derived	This should be 32 bits. These are ignored when widthAdapterSupported = false
wDataMin	64		Derived	This should be 32 bits. These are ignored when widthAdapterSupported = false
wrEn	true	True/False	Always true	
widthAdaptionSupported	false		Derived	
wrapSupported	false		FALSE	

19.10.2. Atui_axi parameters

CSR column describes the value if the CSR would need different value from default parameter

TABLE 19-13: ATUI_AXI BLOCK PARAMETERS

Parameter	Default	Ranges	CSR Network	Description
axiPipeAr	2	0,1,2		User settable
axiPipeAw	2	0,1,2		User settable
axiPipeB	2	0,1,2		User settable
axiPipeR	2	0,1,2		User settable
axiPipeW	2	0,1,2		User settable
beatBufferEntries	0	5.2.2		User settable
ctlPipeCtxt	0	0 .. 9		User settable
ctlPipeReq	2	0,1,2		User settable
ctlPipeResp	2	0,1,2		User settable
enDecodeError	False		True	Fixed as True
enPathLookup	False		False	ATUI: fixed as false
enSplitting	False		False	Always True
idCompMask	[False]			Just use bottom bits. Fixed.
maxOutRd	8		2	User settable
maxOutTotal	2		2	User settable
maxOutWr	8		2	User settable
pipeLevel	2	0,1,2	0	User settable
pipeLevelAtp	2	0,1,2	2	User settable
pipeLevelLut	2	0,1,2	0	User settable
pipeLevelPam	0	0 to log2(maxPAMEntries)	2	User settable
pipeLevelRob	2	0,1,2	0	User settable
pmonStatsEn	False	True/False	False	User settable
rateLmtBktGlobal	8			User settable
rateLmtBktQueue_p	[0]			Fixed
rateLmtBktQueue_s	[0]			Fixed
rateLmtEn	False	True/False	False	User settable
rateLmtRefCntGlobal	8			User settable
rateLmtUseExternalValues	False			Fixed 1
refreshAmtGlobal	8			User settable
refreshAmtQueue_p	[0]			Fixed
refreshAmtQueue_s	[0]			Fixed
reorderingEntries	2		0	User settable

Parameter	Default	Ranges	CSR Network	Description
strpFunc	[' 0']		1	At R1, only struFunc = 1 will be used. Derived
timeoutErrCount	0	5.1.2	0	User settable
wRateLmtBktGlobal	0			Fixed
wRateLmtBktQueue	16			Fixed
wRateLmtRefCntGlobal	0			Fixed
wRateLmtRefCntQueue	16			Fixed
wRefreshAmtGlobal	0			Fixed
wRefreshAmtQueue	16			Fixed

19.10.3. APB socket parameters

TABLE 19-14: APB SOCKET PARAMETERS

Parameter Name	Default	Range	CSR Network	Description/Comment
wData	32	8, 16, 32, 64	32	
wAddr	12	minimum: 12 maximum: 64	12	This can the packet field width can be 12, because once a packet is headed toward a block on the CSR network, only the bottom 12 bits are needed. Bits above 12 are needed to select the block.
wPSel	1	1, 2, 4, 8, 16	1	
wStrb	0	APB2: 0 APB3: wData/8	wData/8	
wPSlverr	0	APB2: 0 APB3: 0 or 1	1	
wProt	0	APB2: 0 APB3: 3	3	
csrAccessSupported	True	True/False	False	
readSupported	True	Fixed true	True	
writeSupported	True	Fixed true	True	

19.10.4. AXI socket parameters

TABLE 19-15: AXI SOCKET PARAMETERS

Parameter Name	Default	Range	CSR network	Description/Comment
wAddr	32	Minimum: 12 Maximum: 64	24	
wArUser	0	Minimum:0 Maximum: 64	0	
wArID	1	Minimum:1 Maximum:32	0	
wAwUser	0	Minimum:0 Maximum: 64	0	
wAwId	1	Minimum:1 Maximum:32	0	
wWUser	0	(wData/8 * 0, 1, 2, 3, 4, and 5) wWUser should be provided not the above but it is actual width. For example, if the data width is 64, and the user bit per byte is 1, wWUser should be 8. if the writeSuported = 0, wWuser = 0	0	
wData	32	8, 16, 32, 64, 128, 256, 512, 1K, 2K	32	
wRuser	0	(wData/8 * 0, 1, 2, 3, 4, and 5) wRUser should be provided not the above but it is actual width. For example, if the data width is 64, and the user bit per byte is 1, wRUser should be 8. if the readSuported = 0, wRuser = 0	0	
wBuser	0	Minimum:0 Maximum: 64	0	
wLen	8	AXI3: 4 [3:0] AXI4: 8 [7:0]	4	
wSize	3	Minimum:3 Maximum:4	3	
wLock	1	AXI3: 2 [1:0] AXI4: 1	1	
wProt	3	3 [2:0]	3	
wQos	4	AXI3: 0 AXI4: 4	0	

Parameter Name	Default	Range	CSR network	Description/Comment
wRegion	0	It is only in AXI4: Minimum:0 Maximum:4	0	
nativeType	Axi4	Axi3/Axi4	Axi4	
eAr	1	0,1	1	
eAw	1	0,1	1	
csrAccessSupported	true	True/False	False	
wrapSupported	false	True/False	False	
narrowSupported	false	True/False	False	
fixedSupported	false	True/False	False	
readSupported	True	True/False	True	
writeSupported	True	True/False	True	
readInterleaveSupported	False	True/False	False	
earlyWriteReponseSupported	False	True/False	False	
maxBurstLength	16	2^N with N = {0 .. 12}	1	

19.10.5. Switch parameters

Network parameters will be fixed. Also, block parameters for all the switches will be fixed

- Only packet parallel style supported at NCore 3.2.
- Only sym_buf_switch will be used.

TABLE 19-16: SWITCH BLOCK PARAMETERS

Parameter		CSR	
defaultArbPolicy	sym_buf_switch	arb_rr1	
defaultInputSwitchDepth	sym_buf_switch	2	BufLayer0.depth
defaultOutputSwitchDepth	sym_buf_switch	0	BufLayer2.depth

20. *Other User Settable Parameters*

20.1. Parameter related with Placeholder Generic Signal

TABLE 20-1: PARAMETER FOR GENERIC PORT: WIRENAME

Name: wire**Name**					Visibility: User
	Architecture		**Release**		**Default**
Value					
Constraints					
Customer Description	portName for Generic port				
Engineering Description					

TABLE 20-2: PARAMETER FOR GENERIC PORT: WIREWIDTH

Name: wireWidth					Visibility: User
	Architecture		**Release**		**Default**
Value					
Constraints					
Customer Description	Port width for generic port				
Engineering Description					

TABLE 20-3: PARAMETER FOR GENERIC PORT: WIRERTLPREFIX

Name: **wireRtlPrefix**					Visibility: User
	Architecture		**Release**		**Default**
Value					
Constraints					
Customer Description	RTL Prefix. For a given block, all the ports must have the same witeRtlPrefix.				
Engineering Description					

TABLE 20-4: PARAMETER FOR GENERIC PORT: DIRECTION

Name: direction					Visibility: User
	Architecture		**Release**		**Default**
	Valid Values		*Valid Values*		
Value	[In, Out]				In
Constraints					
Customer Description	Parameter for port direction configuration				
Engineering Description					

20.2. Parameter related with SRAM assignment

20.2.1. SW_memory

TABLE 20-5: MEMORY PARAMETER FOR IOAIU

Memory Name	Constraints	Number of Memories
OttMem	MemoryProtectionType (Table 4-16) cannot be "NONE" memoryType must be SRAM	Same as #.of nOttDataBanks (Chapter 14.1)

TABLE 20-6: MEMORY PARAMETER FOR SNOOP FILTER

Memory Name	Constraints	Number of Memories
TagMem	MemoryProtectionType (Table 4-16) cannot be "NONE" memoryType must be SRAM	Same as #.of nWays (Chapter 10.3) - bitEn == 0 in NCore 3.2.

TABLE 20-7: MEMORY PARAMETER FOR DMI

Memory Name	Constraints	Number of Memories
writeDataMem	MemoryProtectionType (Table 4-16) cannot be "NONE" memoryType must be SRAM	Same as # of nCohWrDataBanks (Chapter 11.1)

TABLE 20-8: MEMORY PARAMETER FOR CCP

Memory Name	Constraints	Number of Memories
TagMem	MemoryProtectionType (Table 4-16) cannot be "NONE" memoryType must be SRAM	Same as #.of nTagBanks
DataMem	MemoryProtectionType (Table 4-16Table 4-17) cannot be "NONE" memoryType must be SRAM	Same as #.of nDataBanks

TABLE 20-9: MEMORY PARAMETER FOR DVE

Memory Name	Constraints	Number of Memories
TraceMem	MemoryProtectionType (Table 4-16) cannot be "NONE" MemoryType must be SRAM	2

20.2.2. Generic ports

Generic ports are used to create user defined signals for SRAM interfaces. This is a common for all blocks.

Software supports definition of N generic ports of width m ≤ 1023 bits for each block that instantiates memories - need to check if this is implemented, how is it verified, ports created and verified connected all the way through the hierarchy - port reaches all the way to the top level, DFT chimney for DFT signals - user instantiates memory wrapper with his DFT logic - these signals will be used to bring these signals up - need to use the same name as the ports on the memory wrapper etc.

Commented [MF1]:

Commented [MF2]:

TABLE 20-10: PARAMETER FOR SRAM GENERIC PORT: WIRENAME

Name: wireName			Visibility: User
	Architecture	**Release**	**Default**
Value			
Constraints			
Customer Description	portName for Generic port		
Engineering Description			

TABLE 20-11: PARAMETER FOR SRAM GENERIC PORT: WIREWIDTH

Name: wireWidth			Visibility: User
	Architecture	**Release**	**Default**
Value		Max: 1024	
Constraints			
Customer Description	Port width for generic port Maximum width per signal in generic interface is 1024		
Engineering Description			

TABLE 20-12: PARAMETER FOR SRAM GENERIC PORT: DIRECTION

Name: direction			Visibility: User
	Architecture	**Release**	**Default**
	Valid Values	*Valid Values*	
Value	[In, Out]		In
Constraints			
Customer Description			
Engineering Description			

21. User Settable parameter for Synthesis

TABLE 21-1: SYNTHESIS PARAMETER: CHECKONLY

Name: checkOnly		Type: Boolean	Visibility: User
	Architecture	**Release**	**Default**
Value			
Constraints			
Customer Description	Whether to stop the RTL flow after compilation and linking, or to proceed to synthesis		
Engineering Description			

TABLE 21-2: SYNTHESIS PARAMETER: TOPOMODE

Name: topoMode		Type: Boolean	Visibility: User
	Architecture	**Release**	**Default**
Value			
Constraints			
Customer Description	Whether to launch the synthesis tools in topographical mode, or WLM. The latter is faster but less accurate		
Engineering Description			

TABLE 21-3: SYNTHESIS PARAMETER: TECHNOLOGY

Name: technode			Visibility: User
	Architecture	**Release**	**Default**
	Valid Values	*Valid Values*	
Value	ERROR,TSMC16,TSMC7,CUSTOM		CUSTOM
Constraints			
Customer Description	Custom generates a technology template file which contains the variables which need to be filled in with the users technology library information before running synthesis		
Engineering Description			

TABLE 21-4: SYNTHESIS PARAMETER: CLOCKUNCERTAINTY

Name: clockUncertainty					Visibility: User
	Architecture		**Release**		**Default**
	Min	*Max*			
Value	1	99			15
Constraints					
Customer Description	The default clock uncertainty to assume for clocks, as a percentage (e.g. 15 = 15%). The value can be overwritten in the generated synthesis scripts.				
Engineering Description					

TABLE 21-5: SYNTHESIS PARAMETER: RTLWRAPPERDIR

Name: rtlWrapperDir			Visibility: User
	Architecture	**Release**	**Default**
Value			
Constraints			
Customer Description	Directory with user-written Verilog files which instantiate custom cells, such as memories. They override generic-behavior Verilog files generated by Maestro (which implement memories as a \"sea of registers\") and must be named identically.		
Engineering Description			

TABLE 21-6: SYNTHESIS PARAMETER: HARDMACRODBS

Name: hardMacroDbs			Visibility: User
	Architecture	**Release**	**Default**
Value			
Constraints			
Customer Description	Specifies the location and names of the hard macros in the design, such as compiled memories		
Engineering Description			

TABLE 21-7: SYNTHESIS PARAMETER: BOTTOMUPSYNTHESIS

Name: bottomUpSynthesis			Visibility: User
	Architecture	**Release**	**Default**
Value			True
Constraints			
Customer Description	If selected will write out scripts and hierarchy for a bottom of synthesis run, with all the Ncore units written out to their own directories. If not selected a top down, flat synthesis hierarchy will be gnerated.		
Engineering Description			

TABLE 21-8: SYNTHESIS PARAMETER: MAXTRANSITION

Name: **maxTransition**			Visibility: User
	Architecture	**Release**	**Default**
Value			150
Constraints			
Customer Description	Default transition delay on functional input ports (THIS SHOULD BE RENAMED)		
Engineering Description			

TABLE 21-9: SYNTHESIS PARAMETER: OUTPUT LOAD

Name: outputLoad			Visibility: User
	Architecture	**Release**	**Default**
Value			100000
Constraints			
Customer Description	The default capacitive load on functional output ports		
Engineering Description			

TABLE 21-10: SYNTHESIS PARAMETER: ULVTPERCENTAGE

Name: ulvtPercentage			Visibility: User
	Architecture	**Release**	**Default**
Value			
Constraints			
Customer Description	Ulvt Percentage: Ulvt percentage limit set in synthesis scripts.		
Engineering Description			

TABLE 21-11: SYNTHESIS PARAMETER: COMPILECOMMAND

Name: compileCommand			Visibility: User
	Architecture	**Release**	**Default**
Value			
Constraints			
Customer Description	Compile command: Allows the user to add in options to default synthesis command		
Engineering Description			