Ncore 3.4 - Width-/Rate-Adapter Specification

Release: 3.4.23

Rev: 0.5, February 7, 2022

Release Information

Version	Editor	Change	Date
0.1	MF/MK	Initial Document template created	10/24/2019
0.2	MF	Document started from template	02/04/2022
0.5	MF	Added details for each item and cleaned up	02/07/2022
Legend:	MK	Mohammed Khaleeluddin	
	MF	Michael Frank	
	JU	Junie Um	
	KJ	Kjeld Svendsen	
	Xx	Whoever else edited this document	

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.

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.

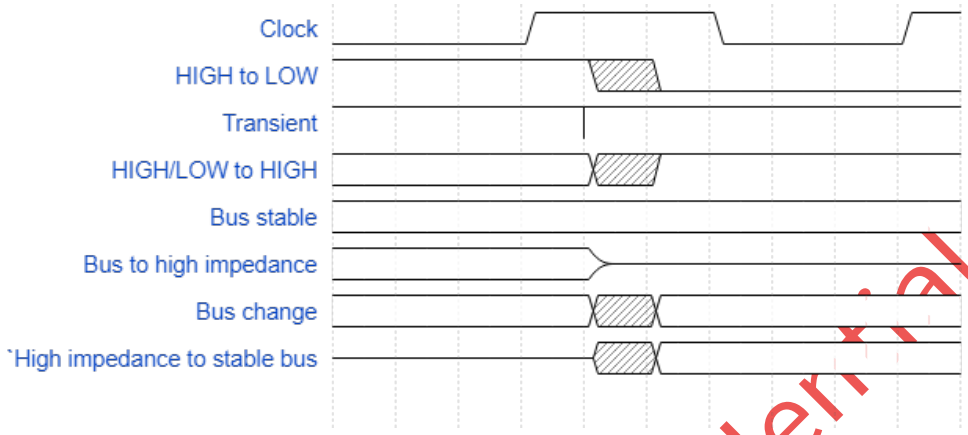
SMALL CAPITALS

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

Flit

A flit (flow control units) is a unit amount of data when the message is transmitting in link-level. The flit can be accepted or rejected at the receiver side based on the flow control protocol and the size of the receive buffer. The mechanism of link-level flow control (e.g. valid/ready) is allowing the receiver signal to the tramsmitter if it should keep sending flits or stop sending flits.

Phit

Every network has a width w, and a transmission rate f, which decide the bandwidth of a network as b = w*f. The amount of data transferred in a single cycle is called a physical unit or phit. As is observable, the width of a network is also equal to the phit size. Hence the bandwidth of the network can also be defined in terms of phit/sec.

Bubble

Clock cycle during which no transfer occurs on a network connection

2. Requirements

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 latency m:

- m = (nOutputWidth/nInputWidth) + ord(boolPipeline) If pipelining enabled

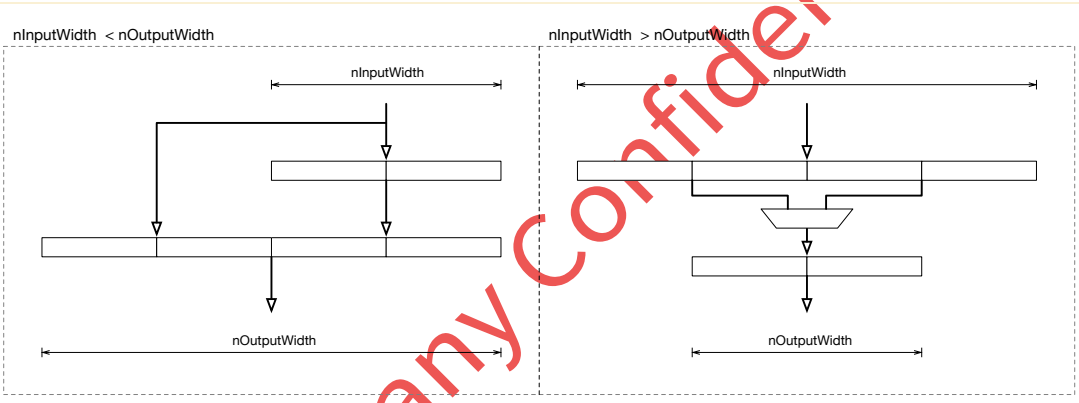


FIGURE 1: INTERNAL STRUCTURE OF WIDTHADAPTER

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

Parameters (WidthAdapter)

These parameters are required to configure the WidthAdapter:

Name: nInputWidth					**Visibility:** none
	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				

Name: nOutputWidth					**Visibility:** none
	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, it is based on the width of the sink connected to this block. Maestro derives this parameter from the {FUnit, Switch}.receiver.width connected to the output of the WidthAdapter				

Name: boolPipeline					**Visibility:** User
	Architecture		**Release**		**Default**
	min.	max.	min.	max.	
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 stage into the WidthAdapter. The setting has no effect if nDepth > 0				

Name: nDepth					**Visibility:** User
	Architecture		**Release**		**Default**
	min.	max.	min.	max.	
Value	0	4 x nTxnSize[1] / nOutputWidth	0	4 x nTxnSize[1] / nOutputWidth	False
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:	1. TxnSize = 64 Bytes = 512 bits				

RateAdapter

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

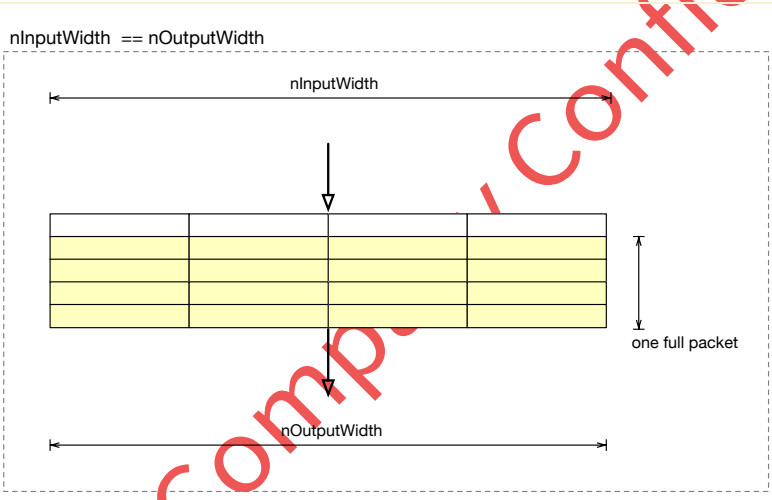


FIGURE 2: INTERNAL STRUCTURE OF RATE ADAPTER

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

Combined Width/RateAdapter

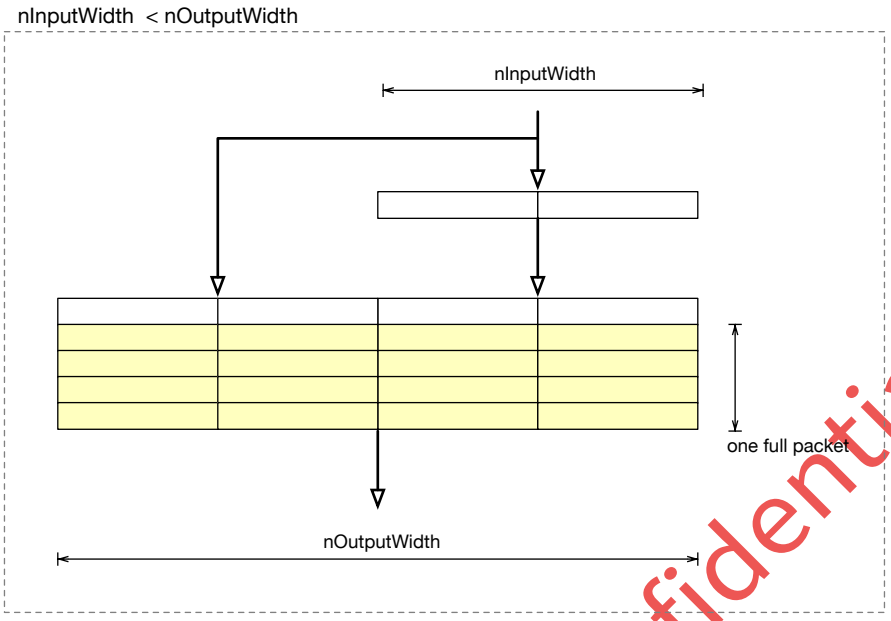


FIGURE 3: WIDTHADAPTER WITH RATEADAPTATION

Figure 3 shows an example of a WidthAdapter that has been parametrized to add buffer stages

Parameters (RateAdapter)

These parameters are required to configure the RateAdapter:

Name: nWidth					**Visibility:** none
	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				

Name: nDepth					**Visibility:** User
	Architecture		**Release**		**Default**
	min.	max.	min.	max.	
Value	2	$\frac{4 \times nTxnSize^{1}}{nWidth}$	2	$\frac{4 \times nTxnSize^{1}}{nOutputWidth}$	$1 + \frac{4 \times nTxnSize^{1}}{nOutputWidth}$
Constraints	nDepth ≥ nTxnSize				
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:	1. 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

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 Input	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 • boolPipelined • nDepth[1]	**User** • Insertion • nWidth • nDepth • boolPipelined	**User** • boolPipelined • nDepth[1]	
nDepth	**Automatic** 1 x nInputWidth + 1 x nOutputWidth **User** + n x nOutputWidth	**User** parameter based on rate difference ≥ n x nWidth	**Automatic** ≥ 1 x nInputWidth **User** + n x nOutputWidth	Automatic insertion will alway 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			

How to determine the configuration parameters

TBD - we will provide user guidance how to determine the number of buffer stages inserted, this will be a function of the difference between arriving traffic and the capabilities of the network segment where the RateAdapter is inserted