ARTERIS IP

Ncore 3 User Guide

ARTERIS.COM

Doc Name	Revision	Date	Description
100023_UG_3.2.1	3.2.1	Jun 24, 2022	Ncore 3, Version 3.2.1
100023_UG_3.2.0	3.2.0	Apr 22, 2022	Ncore 3, Version 3.2.0
100023_UG_3.2_RC2	RC2	Dec 10, 2021	Ncore 3, Version 3.2.0_RC2
100023_UG_3.2_RC1	RC1	Nov 19, 2021	Ncore 3, Version 3.2.0_RC1
100023_UG_1.0_Beta3	Beta3	Oct 13, 2021	Beta3 release
100023_UG_1.0_Beta2	Beta2	Sep 10, 2021	Beta2 release
100023_UG_1.0_Beta1	Beta1	June 15, 2021	Beta1 release
100023_UG_1.0_A.2_D1	1.0_A.2_D1	Mar 12, 2021	1.0_A.2_D1 preliminary release

Contents

1 Introduction

Ncore® is a software platform that enables users to specify, design, validate, and generate fully configurable Arteris interconnects. The software provides two primary interfaces:

- Tcl Interface — Described in the *Tcl User Guide*.
- Maestro® Graphical User Interface — Described in this document.

These interfaces enable you to implement a multi-step process for generating final design collaterals. Ncore enables users to specify and generate the following products:

- Ncore 3 (and higher) cache coherent interconnects.

Ncore is a configurable interconnect IP product that enables the integration of heterogeneous coherent agents and legacy non-coherent agents into complex SoCs.

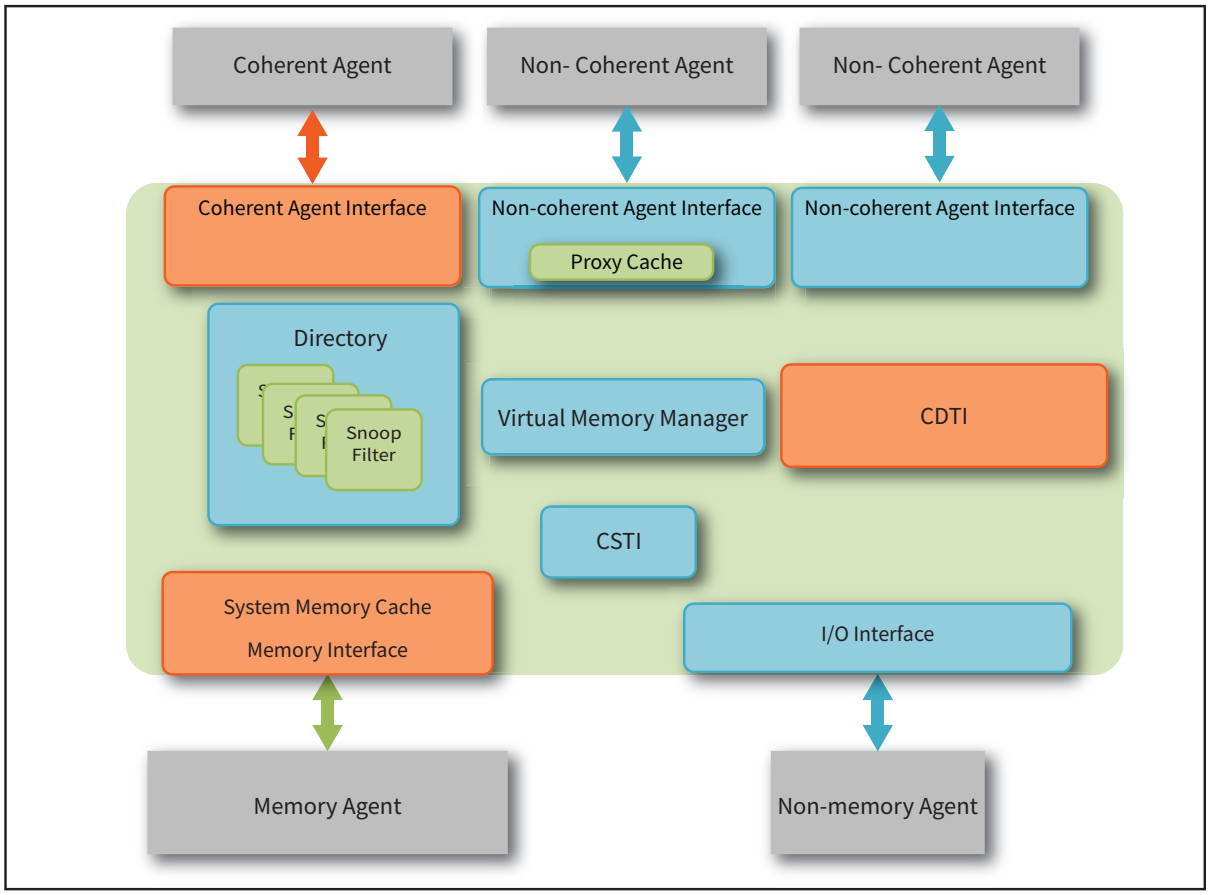
The Ncore cache coherent interconnect supports both the AMBA® ACE and CHI cache coherent protocols. The modular and distributed architecture of Ncore results in higher performance, lower power consumption, and a smaller die area.

The Ncore framework provides the Graphical User Interface (GUI) used to configure the Ncore interconnect. This document describes how to use the Ncore tool, including how to configure the cache coherent and non-cache coherent portions of the interconnect. See the *Ncore Reference Manual* for a detailed description of the Ncore interconnect architecture and hardware features.

Figure 1 shows a logical view of the Ncore interconnect architecture.

Note This document contains a "Maestro Demonstration Design" which goes through the design phases described in "Ncore Workflow Phases"). Some users of Maestro will find it more instructive to run the demonstration design before reading the conceptual information in this document.

Figure 1. Ncore Functional View



1.1 Features

The Ncore interconnect supports an extensive set of features and configurability to allow you to optimize the interconnect to system use cases and requirements.

The Ncore interconnect features include:

- A configurable number of heterogeneous agents (fully coherent, IO-coherent and non-coherent) and a configurable number of Command and Data Transport Interconnects (CDTI).
- Multiple configurable snoop filters that are configurable by size and agent assignment.
- Support for native coherence protocols: CHI and ACE.
- Heterogeneous coherence models, including cache state models, transaction processing models, and transaction types.
- Proxy caches optionally configured in Non-Coherent Agent Interface Units (NCAIUs).
- UVM simulation suite output for verification of the configured interconnect instance.
- SystemC standalone.
- SystemC model output for use within SystemC modeling environments.
- Configurable SMC, called System Memory Cache in Ncore, provides the benefits of a Last Level Cache (LLC) while using less power and area.
- The optional Ncore Resilience Package provides additional functional safety features for ISO 26262 qualification, including:
  - Configurable link data protection
  - Hardware unit duplication
  - Memory protection
  - Functional safety checker IP
  - Fault controller unit
- The number of ports for Distributed Memory Interface Units (DMIUs) and the Distributed Coherence Engines (DCEs) can have configurable values.

1.2 Scalability

The Ncore interconnect supports:

- A configurable number of fully-coherent agent interfaces using the AMBA 4 ACE (AXI Coherency Extensions) protocol.
- A configurable number of IO-coherent agent interfaces using the AMBA4 ACE-Lite and ACE-Lite-E protocols.
- A configurable number of Non-Coherent Agent Interface Units (NCAIUs) for connection to non-coherent subsystems based on Arteris FlexNoC.

- A configurable number of coherent memory regions

1.3 About the Project File

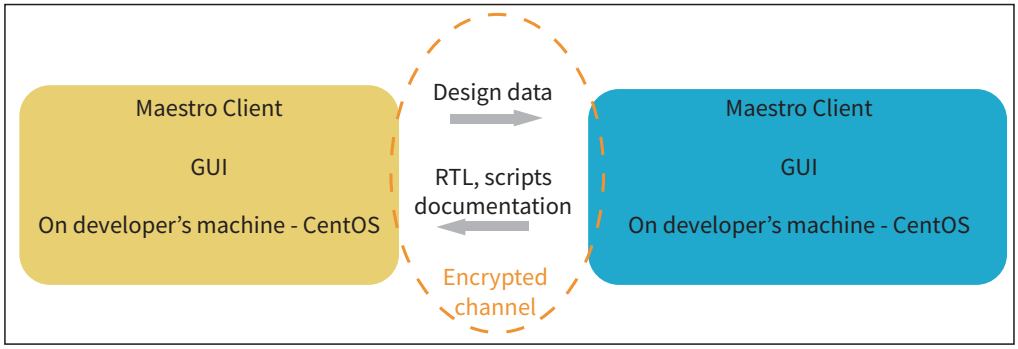
Ncore stores each design in a Maestro Project File (MPF). The file contains all of the design data required by Ncore for a specific project.

1.4 Design Configuration Workflow

The Ncore design workflow is driven by the software cockpit named Maestro. Maestro comprises a client component and a server component.

- The client component includes all designer-visible functionality, including a GUI (see Figure 2). The client oversees specification entry, design definition, design refinement, and design checking.
- The server component, which can be run at Arteris or on-premises depending on the license agreements, is invoked during the design workflow to generate RTL and collaterals, which are then downloaded onto the client's file system.

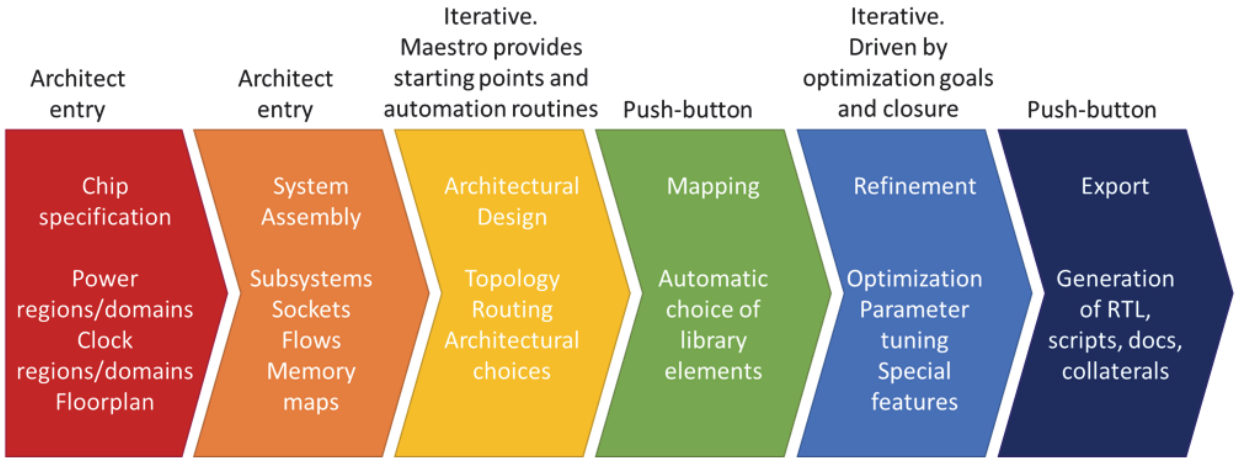
Figure 2. Maestro client-server component



1.4.1 Ncore Workflow Phases

The Ncore workflow encompasses a sequence of design actions known as phases. You perform the design actions and are assisted by Maestro (see Figure 3).

Figure 3. Ncore Design Workflow Phases



1. **Phase 1** - The SoC Specification is the phase of the design flow that defines a chip, power domains and regions, clock domains and subdomains, and floorplan inputs. See "SoC Specification (Phase 1)".
2. **Phase 2** - The System Assembly phase of the design flow defines sockets, connectivity requirements and memory maps, and (in upcoming releases) subsystem composition and decomposition. See "System Assembly (Phase 2)".
3. **Phase 3** - The Architectural (Structural) Design phase of the design flow is where one or more topology solutions are created to meet the performance requirements of the design. The phase can be achieved in either a fully manual method, or the Architectural design can be assisted by Maestro. For example, see the Structural Design chapter for information about how Maestro can create a fully-connected regular topology to realize the connectivity. This design phase includes creation and configuration of Ncore units to offer cache coherence. See "Structural Design (Architecture, Phase 3)".
4. **Phase 4** - Mapping is the phase of the design flow that casts the solution(s) of the previous point into specific versions of specific components in the Arteris IP library. Currently, this step is fully automatic. See "Mapping (Phase 4)".
5. **Phase 5** - Architectural Refinement is the phase of the design flow during which the designer can fine-tune parameters to achieve specific PPA objectives, for example, choosing buffer sizes, tweaking arbitration policies, deploying pipe stages to achieve timing closure, etc. See "Architectural Refinement (Phase 5)".
6. **Phase 6** - Export is the phase of the design flow that invokes the server component of Maestro to generate RTL and collaterals. This process is only allowed if the design is clear of any outstanding errors. See "Export Design (Phase 6)".

Throughout the design flow, Maestro provides guidance to the designer about the current and next steps. Several actions are automated to minimize the iteration time and the possibility of misconfiguration. Additionally, checks are performed in the flow to prevent incorrect parameterizations.

2 Graphical User Interface

This chapter describes the tools and controls in the Maestro Graphical User Interface (GUI).

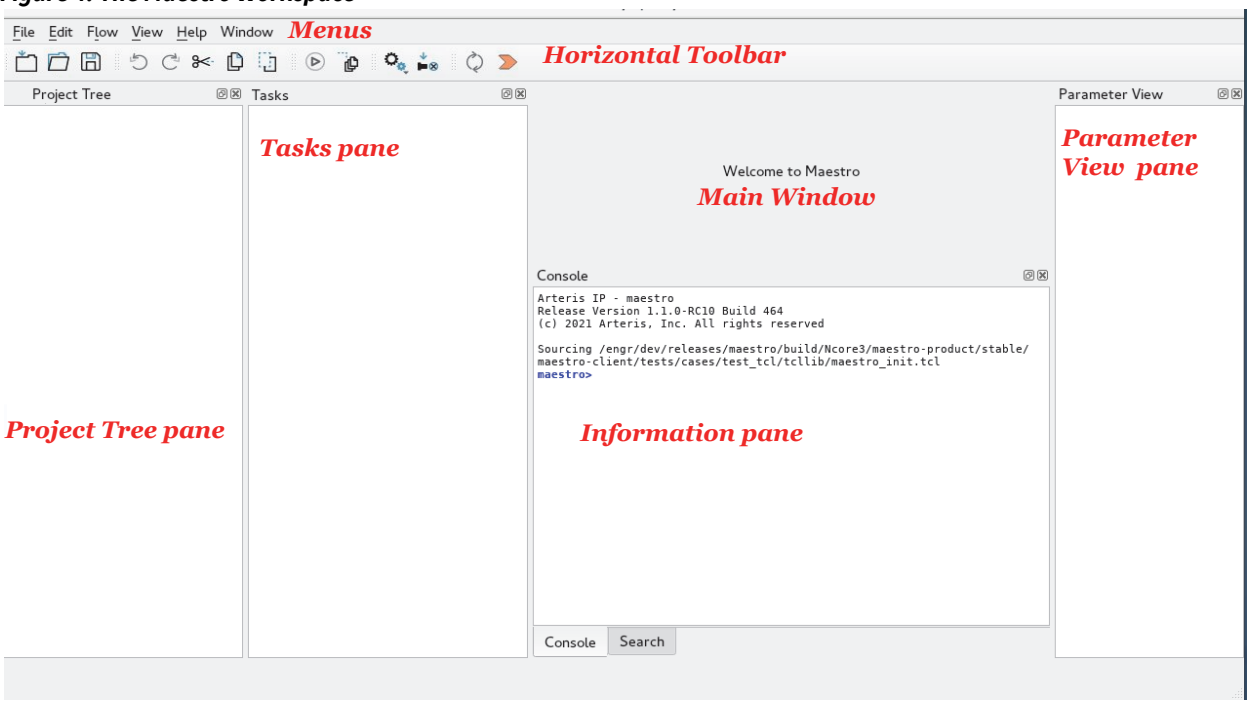
Maestro software is the complete software environment provided to the user when they license an ArterisIP product. The terms Maestro software, Maestro cockpit, or Maestro, will be used to describe the Maestro software.

2.1 Workspaces

The initial view of the Maestro workspace is shown in Figure 4. This view can be divided as follows:

- "Horizontal Toolbar"
- "Flow Navigator Ribbon, Tasks Pane, and Project Tree Panes"
- "Main Window"
- "Information Pane"
- "Parameters Pane"
- "Menus and Shortcuts"

Figure 4. The Maestro Workspace



The arrangement of the panes may be different, depending on your settings. Panes can be docked/undocked by using the dock/undock icon located in the upper-right of each pane.

2.1.1 Horizontal Toolbar

This section explains the functions of the controls and icons provided in the horizontal toolbar.

Figure 5. Maestro Horizontal Toolbar

Table 1. Toolbar Icon descriptions

Key	Icon	Name	Description
A		New project	Select this icon to open a screen where you can create a new project.
B		Open project	Open a screen where you can navigate to an existing project.

Table 1. Toolbar Icon descriptions

Key	Icon	Name	Description
C		Save project	Saves the open project to the open MPF file. If the project is new, Maestro software prompts you to enter a file name and to select a destination directory for the file.
D		Undo	Undo an action in the project.
E		Redo	Redo an action in the project.
F		Cut	Cut the selected object.
G		Copy	Copy the selected object.
H		Paste	Paste an object that has been copied.
I		Save & Execute	Save & Execute (for Tcl files).
J		Generate RTL	Generate RTL for the design.
K		Automaton	Access sub menus: - Topology wizard - Insert Power Management - Insert Interrupt Handling - Insert Resiliency Handling - Insert Configuration Network - Insert Adapters
L		Insert interface units	Insert interface units.
M		Redraw	Force a redraw of the GUI.
N		Flow Navigator Toggle	Toggles the horizontal ribbon which shows the flow phases of the design.

2.1.2 Flow Navigator Ribbon, Tasks Pane, and Project Tree Panes

Flow Navigator Ribbon

Use the Flow Navigator Toggle icon in the toolbar to display a ribbon which shows the current design phase.

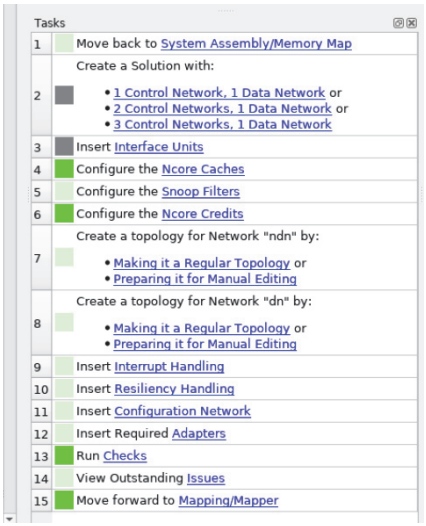
Each flow stage is a high-level workflow, equivalent to a Maestro Phase which contains a set of tasks. Toggle the icon to hide the ribbon.

Tasks View Pane

The figure below shows the various subsections within the Tasks View Pane of a particular Maestro Phase (Structural Design). Tasks are sub-units of phases, and provide an incremental guide to sequentially completing a phase.

You can move back to a previous phase when in Tasks View, if necessary. When you are done with a phase, you move forward to the next phase after you run checks and make sure there are no outstanding issues.

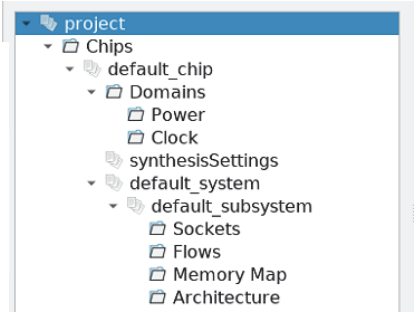
Figure 6. Tasks View



Project Tree Pane

The figure below shows the various subsections within the Project Tree pane. This view allows you to navigate the entire project, so you can view, select, and change items as needed.

Figure 7. Project Tree



2.1.3 Main Window

Depending on the stage the user is in and the state of the design, the Main Window contains one of several editors or viewers. It also provides a graphical display of the process flow as you progress from start to finish. For example, when you select 'SoC specifications > Domains' in the Flow Navigator, it shows you both Power Regions and Clock Regions. You can now view, edit, and modify these domains as per the design requirement.

2.1.4 Information Pane

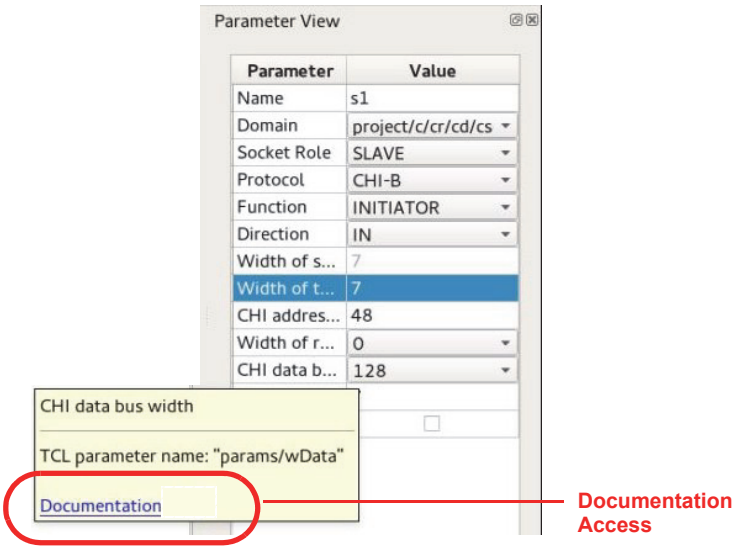
The Information Pane has sections displaying the Tcl Console, Issues (error and warning messages), and status information. For more information, see "Additional UI Features" on page 23.

2.1.5 Parameters Pane

The Parameters Pane allows the user to view or edit parameters appropriate for the object(s) currently selected in the Main Window.

Parameter View Workspace

Use the Parameter View to view and edit parameters.



Parameters can be changed by editing the value on the right. When you mouse over the value, a pop-up window appears, providing access to the documentation for the parameter (click blue link to open Online Documentation).

Parameters are described in *Maestro Parameters*. An alphabetical *Parameter Index* is also available.

2.2 Menus and Shortcuts

- "File Menu"
- "Edit Menu"
- "Flow Menu"
- "View Menu"
- "Help Menu"
- "Window Menu"
- "Additional UI Features"

2.2.1 File Menu

Controls/commands that deal with files and are only available on a text file.

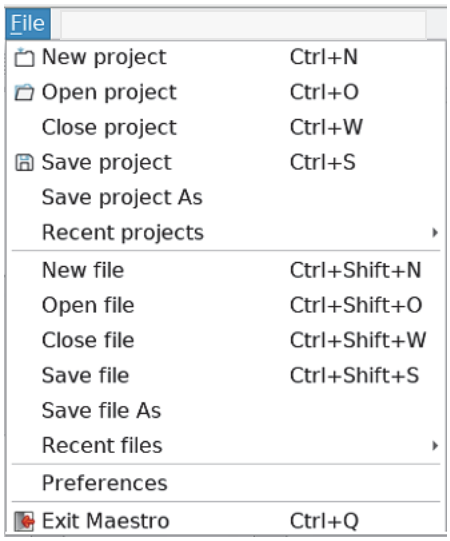


Table 2. File menu descriptions

Control	Description
New project	Create a new Maestro project.
Open project	Open a Maestro project. The project file suffix is *.mpf. You can only have one project open at a time.
Close project	Close the current project.
Save project	Save the current project. Tip: Maestro automatically saves the project when you close the terminal session
Save project As	Save a copy of the project that is currently open. Tip: Maestro automatically saves the project when you close the terminal session.
Recent projects	List recent projects.
New file	Open a text editor in the center window.
Open file	View a list of files in the Maestro directory that can be opened in a text editor.
Close file	Close a text file.
Save file	Save a text file.
Save file As	Save a copy of the text file that is currently open.
Recent files	List recent files.
Preferences	View a page that you can use to edit the properties assigned to the Maestro desktop. For example, you can change the colors of each Maestro component and enable or disable the display of functions.
Exit Maestro	Close the current Maestro terminal session.

2.2.2 Edit Menu

This section explains how to use the controls and icons that are displayed in the Edit menu.

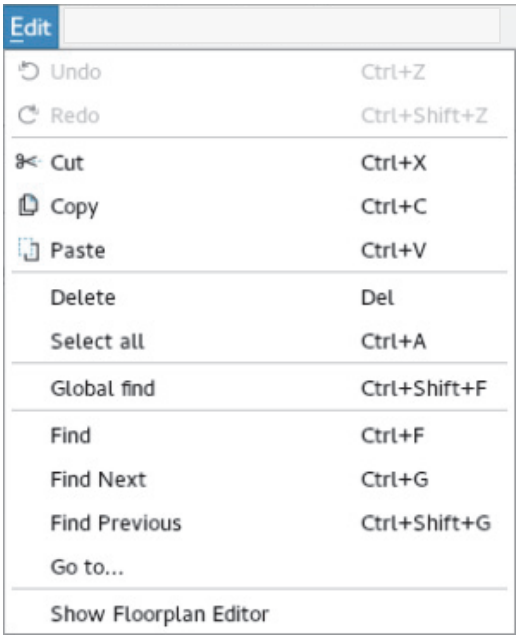


Table 3. Edit menu descriptions

Control	Description
Undo	Undo the previous action.
Redo	Redo the previous action.
Cut	Cut the selected object.
Copy	Copy the selected object.
Paste	Paste the copied object.
Delete	Delete the selected object.
Select all	Selected all objects in a window.
Global find	Find and highlight all search terms in all Arteris databases.
Find	Find a single term in all Arteris databases.
Find Next	Find the next instance of a single term in the Arteris database.
Find Previous	Find the previous of a single term in the Arteris database.
Go to	Go to an object that contains the search term.
Show Floorplan Editor	Show the floorplan editor.

2.2.3 Flow Menu

This section explains how to use the controls and icons that are displayed in the Flow menu.

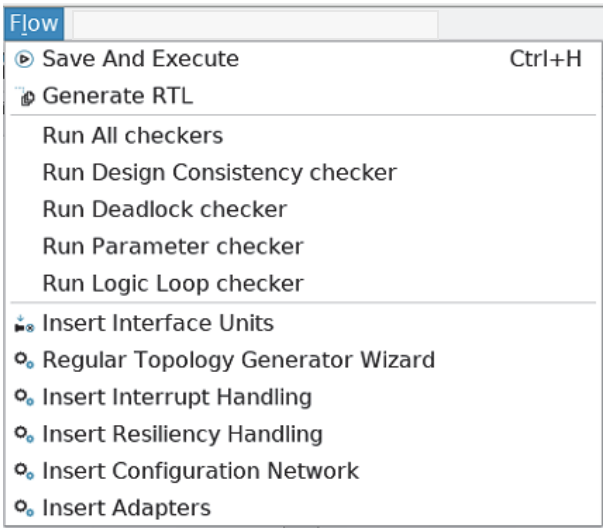


Table 4. Flow menu descriptions

Control	Description
Save and Execute	Save and execute design.
Generate RTL	Generate RTL for the design.
Run All checkers	Run all checkers.
Run Design Consistency checker	Run the Design Consistency Checker.
Run Deadlock checker	Checks for deadlocks. Note: The generated topology should be deadlock free.
Run Parameter checker	Run Parameter Checkers.
Run Logic Loop checker	Run Logic Loop Checker.
Insert Interface Units	Insert interface units.
Regular Topology Generator Wizard	Run the Regular Topology Generator Wizard.
Insert Interrupt Handling	Insert interrupt handling.
Insert Resiliency Handling	Insert resiliency handling.
Insert Configuration Network	Insert configuration network.
Insert Adapters	Insert required adapters.

2.2.4 View Menu

The following table explains the use of controls and icons displayed in the View Menu.

Table 5. View menu descriptions

Control	Description
Redraw	Select this icon to force a redraw of the GUI.

2.2.5 Help Menu

Use the help menu to access the context-sensitive online help, access a guide or manual, release notes, look up a number to contact tech support and learn more about Arteris IP.

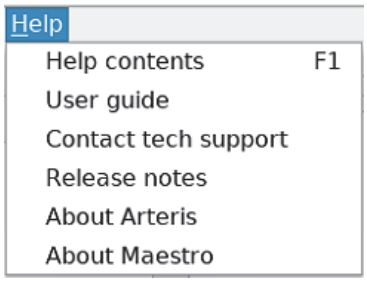
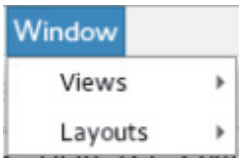


Table 6. Help menu descriptions

Control	Description
Help contents	Open the complete Online Help system.
User guide	Access a PDF version of the Ncore User Guide.
Contact tech support	Access information about how to contact tech support.
Release notes	Access the Ncore Release Notes.
About Arteris	Learn more about Arteris IP.
About Maestro	Learn more about the Maestro software platform.

2.2.6 Window Menu

The Window Menu has selections for Views and Layouts which are the same features in the “Maestro Properties Editor” on page 27. In addition, Window includes selections to view the Source, Topology, and Floorplan editors,



2.2.7 Additional UI Features

Topology Editor Toolbar

The Topology Editor toolbar is visible when using the Topology Editor.

Figure 8. Maestro Topology Editor Toolbar

Table 7. Topology Toolbar Icon descriptions

Key	Icon	Name	Description
A		Toggle Connectivity Table	Toggle connectivity table visibility.
B		Toggle Routes Table	Toggle routes table visibility.
C		Layout to Grid	Move from layout to grid view.
D		Dynamic Layout	Use dynamic layout.
E		Fit View	Fit view to window.
F		Zoom 1:1	Zoom to 1:1 ratio (no zoom in/out).
G		Zoom In	Zoom window in.

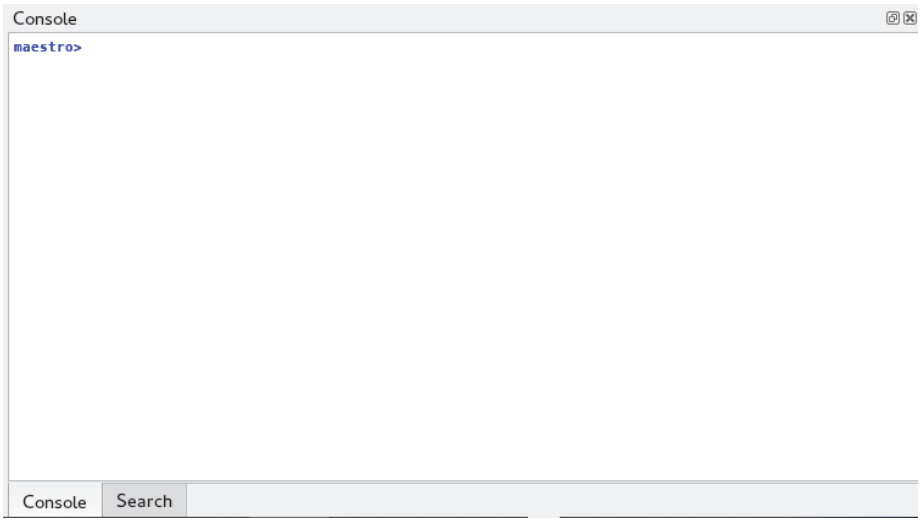
Table 7. Topology Toolbar Icon descriptions

Key	Icon	Name	Description
H		Zoom Out	Zoom window out.
I		Zoom Selected	Zoom the selected object.
J		Prepend Adapter	Prepend an adapter.
K		Prepend Switch	Prepend switch to all inputs of the selected object.
L		Append Adapter	Append an adapter.
M		Append Switch	Append switch to all outputs of the selected object.
N		Insert Adapter	Insert adapter to selected outputs.
O		Split Switch	Split the selected switch.
P		Merge Switch	Merge the selected switches.
Q		Move Routes	Move the selected routes.
R		Delete the Element	Delete the selected element.
S		Toggle Topology View	Toggle topology view configuration panel visibility.

Console interface and Other tabs

This section describes the functions and features of the Console and other tabs. The following figure shows the default Console interface display.

Figure 9. Console Interface



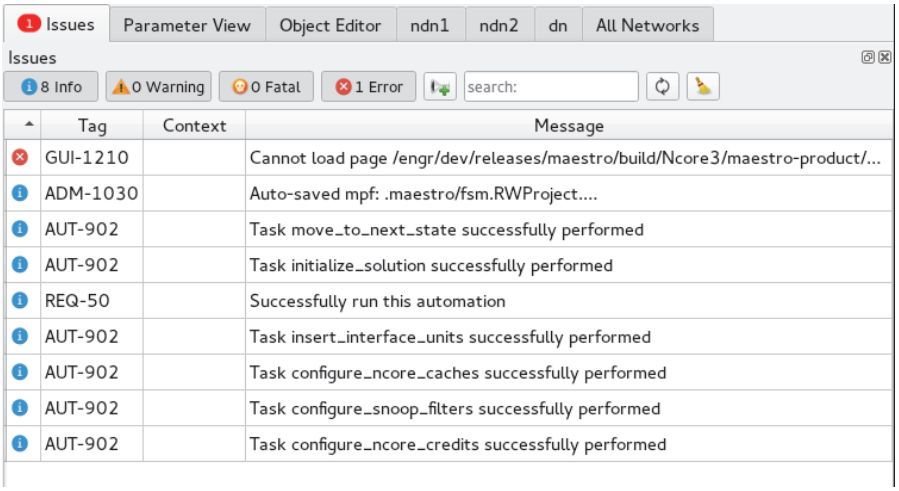
- **Console tab** - Tcl command interface. A Tcl script can be sourced from the Tcl console in the following way: `source <file-name.tcl>`

  When a Tcl script is being run, it displays all the operations taking place to create a design and all other operations from the user are blocked until the script completes. If the script is interrupted before completion, all the work will be lost since the last save.

- **Search tab** - Enter regular-expression terms to locate an item or other information in the Arteris IP database. The search results are displayed in the console window. The results are formatted with a hyperlink to enable direct access to the selected object.

Other Maestro tabs include:

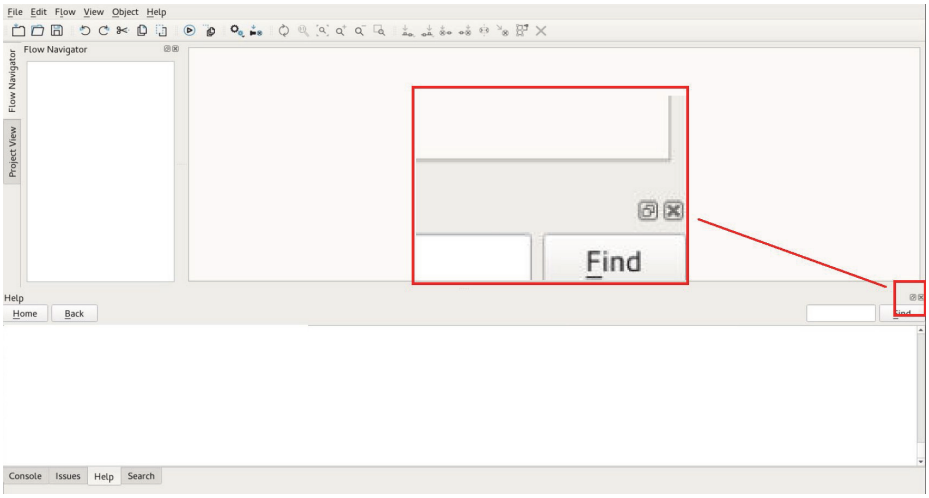
- **Help: Tcl_Command tab** (top left of GUI) - Displays help for Tcl commands.
- **Issues tab** - Displays Information, Warnings, Fatal messages, Deprecated issues, Error messages, and Internal messages



- All Error messages must be corrected before you can proceed to the next design phase.
- To filter by message type, select a button such as Warnings.
- Select the Filter by Tag icon to filter issues.
- To clear all checkboxes in the list of source types, select clear all (sweep icon).
- To set all checkboxes in the list, select select all. Select one or more checkboxes to set the search parameters.

Undock and restore the Maestro interface

1. Click a tab in the Console section of the window. For example, select the Console tab.
2. Select the dockable icon (mid-right above “Find”) .



3. Drag the Console tab to its new location.
4. To restore the default view of each tab in the console, select the top bar of each tab.
5. Release the left mouse button to redock the tab in its default location.

Project Tree pane

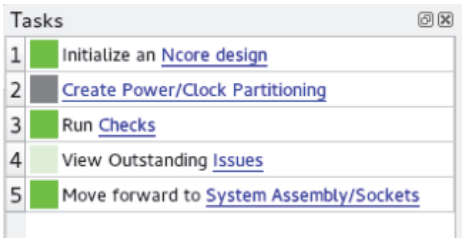
The Project Tree pane displays all of the components in the database.

Flow Navigator ribbon

The Flow Navigator ribbon displays the titles of all project phases.

Tasks pane

The Tasks pane uses green and gray squares to indicate task states withing each phase, as shown here for the SoC Specification phase.



- A green square indicates a task needs to be done.
- A faded green square indicates the task has been performed, but can be performed again (e.g., inserting a socket).
- A gray square indicates a task has been performed and does not need to be done again (e.g., inserting interface handlers).
- If a task with a gray square is clicked, a message in the console appears informing the user that no work for that step needs to be done.

Object Editor pane

The Object Editor displays parameters, values, and data about selected objects in the middle window of the Project Tree. Select the Object Editor tab to view information or close the Object Editor.

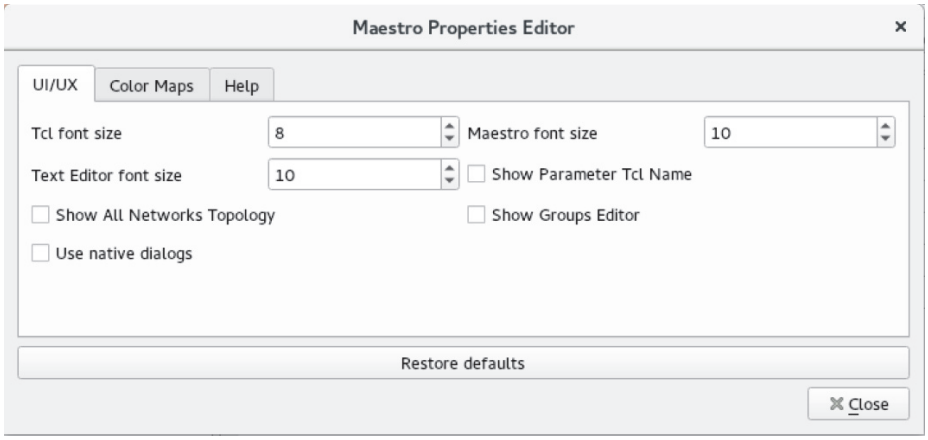
View Parameter pane

The Parameter pane displays parameters, values, data, and other data about the selected objects in the upper-right pane of the Project Tree. Use File > Preferences (Maestro Properties Editor) to restore the Parameter pane, if closed.

Maestro Properties Editor

To view and modify the Maestro GUI related properties, complete the following steps.

1. Select File > Preferences. The Maestro Properties Editor is displayed and shows the UI/UX tab.



2. Use the fields on the UI/UX tab to:

- Change the Tcl font size
- Change the Text Editor font size
- Change the Maestro font size

3. Use the checkboxes to show or hide:

- Parameter Tcl Name - Toggles between value and Tcl name
- Show All Networks Topology
- Use native dialogs
- Show Groups Editor

Select the Restore default bar to restore default UI/UX properties.

Select the Color Maps tab to display options that allow you to configure a custom color palette for a selected parameter, such as a width_Adaptor.

3 SoC Specification (Phase 1)

SoC specification is the first phase of the Maestro design flow. The phase focuses on the power and clock physical specifications used by the system under design.

3.1 SoC Specification Overview

In Phase 1, you will create a chip and then specify the properties of the power and clock distribution within the chip. Because power supply and clock supply are not orthogonal concepts in Maestro, power management handling requires clearly defined relationships between how design components are connected to the power and clock networks.

3.1.1 Chip

The first task in SoC Specification is the creation of a chip. The chip is the container for everything else in the design. A chip correlates to a silicon die.

3.1.2 Power Regions and Domains

Ncore supports a logical hierarchy of power and clock inputs and enables handling of power management.

At the highest level, the chip is subdivided into one or more Power Regions. The main property of a Power Region is the supply voltage. The chip therefore represents a portion of the chip that runs at that voltage.

Power regions are subdivided into one or more Power Domains. The main feature of a Power Domain is to encode whether a portion of the Power Region can or cannot be switched off by disconnecting the power supply.

Note It is possible to divide a Power Region into any number of Power Domains. For uniformity, Maestro requires you to create at least one Power Domain, even if no logic is ever to be gated.

The main purpose for specifying the power regions and domains is to facilitate the need to build the interconnect in a way that allows portions of the system to independently manage power. In addition, specifying power regions and domains allows the interconnect to handle requirements such as the insertion of level shifters through Unified Power Format (UPF) directives.

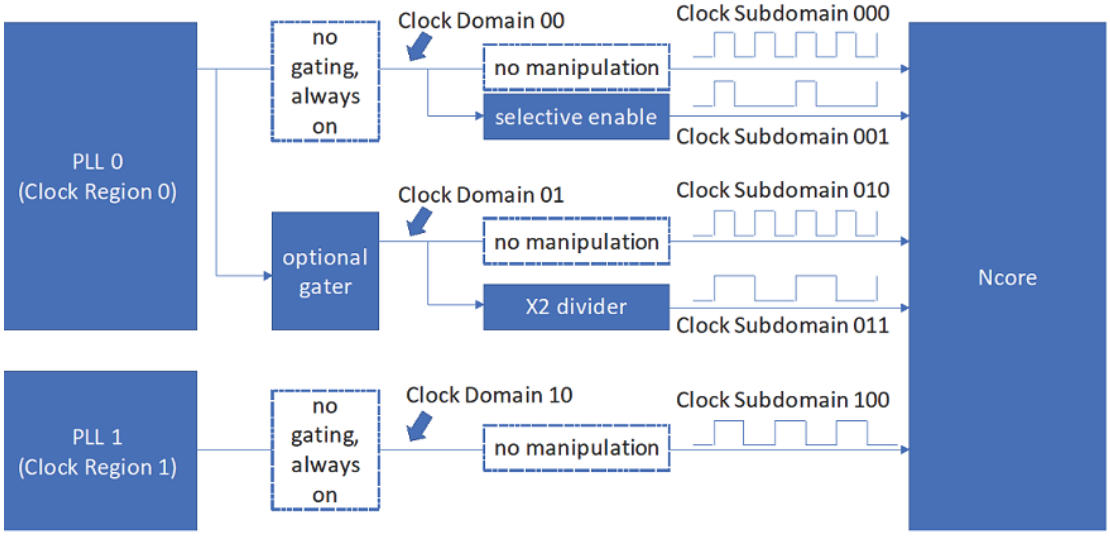
3.1.3 Clock Regions, Clock Domains, and Clock Subdomains

This section provides information about various clocking concepts in Maestro.

The chip is divided into one or more clock regions. A clock region represents a subdivision of the chip that is directly or indirectly fed clock signals that originate from a given clock source, such as a PLL output. A clock region is designed to be entirely contained within a power region. One reason to constrain the clock within a power region is to minimize the possibility of timing jitter caused by fluctuations in the PLL power supply.

Clock regions contain one or more clock domains. A clock domain is a portion of the clock region that might be gated for power management. You can define any number of clock domains. Maestro requires at least one clock domain, even if gating is not specified.

Figure 10. Clock Regions, Domains, Subdomains, and clock signals



Note A clock domain is constrained to be entirely enclosed in a Power Domain.

Clock domains can be divided into one or more clock subdomains. A clock subdomain represents a manipulation of the clock signal. Examples of manipulation include division or edge suppression. A minimum of one clock subdomain must be defined, even if it represents the direct propagation of the clock domain's signal.

Note Ncore does not provide internal clock manipulation logic. For example, there are no gaters or dividers that only provide clock domain crossing logic. Therefore, clock subdomains correspond to the actual clock signals that connect to the system at the RTL level. In comparison, concepts of a clock region and a clock domain only serve to encode the relationships between these clock signals. The direct RTL equivalent is outside the scope of Ncore.

Note Because clock edges can be aligned or mesochronous, optimized crossing logic can be instantiated if the interconnect must span across clock subdomains that descend from a common clock domain. Clock subdomains that descend from independent ancestors always trigger the insertion of dual-clock FIFOs.

Ncore uses the parameters configured in Phase 1 to help ensure that subsequent phases of the design flow are configured to follow the correct clock and power parameters.

Note All instances of Arteris blocks must be attached to at least one clock subdomain. The attachments can be defined automatically or by manual configuration. At least one clock subdomain must be defined before the flow can proceed to Phase 2, which has a small number of blocks, such as those that provide internal clock crossing capabilities, and can connect to multiple clock subdomains.

3.1.4 Before you Begin Phase 1

The instructions in this section are based on the following assumptions:

- The Maestro and Ncore software successfully installed and activated.
- You can access the Maestro Graphical User Interface.
- The appropriate license file(s) are present.

Refer to the *Getting Started Guide* for more information.

About the project names

At the beginning of the project, you create a file named <*projectname*>.mpf in a selected directory. When you exit Maestro, all configuration and data related to the design is saved in this file.

3.1.5 Open Maestro

1. Open the Maestro Graphical User Interface.
2. Use the Flow Navigator toggle to displays a ribbon with the names of the Ncore design phases.

3.1.6 Create a New Project

To create a project:

1. In the Maestro menu bar, select File > New project. The Create Project window is displayed. In the Project Name text box, enter a name for the new project.
2. Enter the Project and File names for your project.
3. In the “Create In” textbox, accept the default directory as the location to store the new project -- or -- select Browse to store the new project in a different directory.
4. Select the License dropdown list to display the names of available license(s), and select a license from the license list.
5. Click Create. The new project is created in the directory selected above.

3.1.7 Create a Chip

1. Assuming you have created a new project using the steps in the previous section, click the “Project View” tab on the left to view the initial Project View.
2. In Tasks View, click “Initialize an Ncore design”. The Project Tree will be populated with a default chip, system, and subsystem.
3. The name of the chip is called "default_chip". To rename it, double-click on "default_chip" in the Project Tree view and enter a new name.

3.1.8 Create a Power and Clock Region

1. Click on the "Create Power/Clock Partitioning" in the Task view.
2. A set of default power and clock regions are automatically created.
3. To change the names, double-click on the name and enter a new name in the Project Tree view.

3.1.9 Create a Power Domain

A Power Domain consists of one Power Region and one Power Domain.

You perform all steps in this section in the Project Tree.

3.1.10 Configure the Power Region Parameters

To configure Power Region parameters:

1. Click on the Power Region name in the Project View.
2. The Object Editor will display the parameters in the main window.
3. Enter a value in the Voltage field. The Voltage units are in mV.
   All views are refreshed with the new parameter values.

3.1.11 Configure the Power Domain Parameters

To configure Power Domain parameters:

1. Click on the Power Domain name in the Project View.
2. The Object Editor will display the parameters in the main window.
3. Enter a value in the Voltage field. The Voltage units are in mV.
   All views are refreshed with the new parameter values.

Power Domain Parameter Descriptions

This section describes the parameter names and values used during Power Domain configuration.

Power Region

A Power Region represents a common property in term of clock and power supply for the logic that it contains. Power Regions are subdivided into domains.

Voltage

Voltage of the power supply of the Power Region

This value represents contains the voltage value or set of values (DVFS) used by the Power Management Unit (PMU) supply.

Power Domain

A Power Domain object represents a subset of a Power Region that contains the set of states into which the domain can be placed. The states are either ON or OFF.

Gating

The gating value determines whether the clock specified in the parent clock region is subject to gating when sent to a portion of the chip, that shall be described as this clock domain. Specify ‘always_on’ if no gating should be applied, or ‘external’ if logic external to the interconnect can gate this clock. If different gating signals can modulate the clock from the same clock region, each must be modeled as a separate clock domain. Ungated propagation shall also be modeled as its own clock domain.

3.1.12 Create a Clock Domain

A clock domain consists of one clock region, one clock domain, and one clock subdomain.

You perform all steps in this section in the Project Tree.

3.1.13 Configure the Clock Region Parameters

A default clock region, domain, and subdomain is created during the initial power region creation.

1. In the Project View, double-click on the clock region, domain, or subdomain to change the name and their parameters in the Object Editor.
2. For example, for the clock region expand the Domains directory.
3. Right-click the clock directory. A menu is displayed.
4. Select Create > Clock Region.
5. Enter the name of the new clock region.
6. Select the down arrow in the Power Region field. A list displays the path to available Power Region(s).
7. Select a path to a Power Region.
8. Enter a value in the Frequency (MHz) text box.

All views are refreshed with the new parameter values.

3.1.14 Create a Clock Domain

To create a clock domain, complete the following steps.

1. Expand the Domains directory.
2. Right-click Clock > clock region. A menu is displayed.
3. Select Create > Clock Domain. The Create Clock Domain window is displayed.
4. Enter the name of the new clock domain.
5. Select OK. The Object Editor and Parameter View display the clock domain name and two additional parameters that require configuration.
6. Select the down arrow in the Power Domain field. A list displays the path to available Power Domains.
7. Select a Power Domain from the list.
8. Select the down arrow in the Gating field. A list containing two values is displayed.
9. Select either always_on or external.

All views are refreshed with the new parameter values.

3.1.15 Create a Clock Subdomain

To configure a clock subdomain, complete the following steps.

1. Right-click on the clock domain name. A menu is displayed.
2. Select Create > Clock Subdomain.
3. Enter a name for the clock subdomain.
4. Select OK. The Object Editor and Parameter View display the clock subdomain name and two additional parameters that require configuration.
5. Click the Divide By field. The field displays up and down arrows.
6. Operate the arrows and select a value that is appropriate for the network design.

7. Enable or disable the checkbox to set the Unit-Level Clock Gating parameter.

All views are refreshed with the new parameter values.

3.1.16 Clock Domain Parameter Descriptions

This section describes the parameter names and values used during clock region configuration.

Clock Region

The name of the clock region.

Power Region

Power Region this object is inside of.

Frequency (MHz)

Frequency of the input clock signal of the clock region.

Clock Domain

The name of the clock domain.

Power Domain

The name of the Power Domain that this object is in.

Gating

Whether the state is always_on or external.

Clock Subdomain

The name of the clock subdomain.

Divide By

Clock divider that is applied to the clock region. The divider applies to the frequency specified in the parent clock region, and subject to the optional gating specified in the parent clock domain. If the clock signal is propagated straight without manipulations, specify 1.

Unit-level Clock Gating

Whether it is desired that units receiving a clock signal from this clock subdomain should be subject to unit-level clock gating, when possible, for power savings. Disabling this optimization might help timing.

Note The instructions in the next section assume that you have a .mpf project file open.

3.2 Create a System and a Subsystem

To create a system and subsystem, complete the following steps. The subsystem is a self-standing portion of the architectural design. The subsystem communicates with the rest of the system by way of sockets and ports.

Restriction: You can create only one cache-coherent Ncore system and one subsystem.

If you created the chip through the Task View task, a default system and subsystem is already added/created which makes these steps unnecessary.

1. Expand Chips <*chip name*>.
2. In the Project Tree, right-click <chip name>. A menu is displayed.
3. Select Create > System.
4. Enter the name of the new system.
5. Select OK. The Object Editor and Parameter View display the system name.
6. Right-click the name of the new system. A menu is displayed.
7. Select Create > Subsystem.
8. Enter a name for new subsystem.
9. Select OK.

The Object Editor and the Parameter View are updated.

3.2.1 SoC Specification Configuration Verification Test

To verify successful completion of Phase 1, complete the following steps.

1. Select the Flow Navigator ribbon.
2. Expand SoC Specification > Domains.
3. Check for any issues to ensure the SoC Specification (Phase 1) was completed successfully.
4. Select the Project Tree.
5. Expand Chips <chip name>.
6. Expand the System directory.
7. Verify that the subsystem directory is present.

Select the Project Tree and complete the following steps.

1. Review the Issues console to detect errors, warnings or messages relating to the SoC specification phase.
2. Correct any error conditions as required.
3. Continue to correct errors until after clicking the “Check for Issues”, the console displays the message that all tasks were cleared successfully.

4 System Assembly (Phase 2)

The Ncore system is composed of logical elements, such as systems and subsystems. This Ncore release supports only one system and one subsystem.

A subsystem is a self-standing portion of the architectural design. The subsystem communicates with the rest of the system by way of sockets and ports. Sockets, for example, AXI and CHI, are the main input and output of the subsystem. Sockets interface with the IP library.

In this phase you will configure sockets and a memory map in the subsystem. Ncore features a highly flexible, runtime-programmable storage architecture. With the exception of the CSR region, (a hard-coded memory address) you are not required to enter memory addresses in this phase.

4.1 System Assembly Socket Overview

This section contains reference and conceptual information about sockets in Ncore.

You must specify a set of interface sockets to connect to your IPs. The following information is required to configure each socket:

- A Protocol.
- A Function. The function of the Socket is to define which type of Ncore Unit is deployed on the inside of the subsystem. Allowable values are:
  - The INITIATOR function can only be assigned to initiators configured with the role of FUNCTION. Selecting the INITIATOR function determines whether the socket is going to be a CAIU or an NCAIU.
  - The MEMORY function can only be assigned to targets connected to memories. Selecting a MEMORY function determines that the socket is a DMI.
  - The PERIPHERAL function can only be assigned to targets that have the role of function and are connected to peripherals. Selecting a PERIPHERAL function determines that the socket is a DII.
- A Clock Subdomain. During socket creation you must select a clock subdomain which encodes the clock at which the socket operates.

More protocol-specific parameter values might require connection during socket creation (see Section “Common Socket Parameter Descriptions”.

Sockets expose the signals they describe as bundles of related ports of the IPG or unit that contain them. For instance, an AMBA/AXI socket describes all the ports in the AXI interface as a bundle of related ports.

Sockets are bound through association with a subdomain clock object. All signals of the socket are synchronized to the selected subdomain clock object. A clock subdomain belongs to a particular clock domain. The clock domain is related to a particular Power Domain. The validity of the signals of the socket follow the same rules as all of the logic inside the Power Domain. If the domain is OFF, output signals are floating.

Socket creation rules include:

- Selecting the type of socket and the protocol.
- Setting parameters according to the requirements of the selected protocol.
- Determining the maximum number of sockets permitted for a selected protocol.
- Consideration of inter-socket dependencies and compatibility

Socket parameters are configured directly by you in this phase.

When two sockets that have configuration information are connected, Maestro checks that the protocol and parameters are compatible.

Each socket in a cache-coherent interconnect is automatically attached to a Network Interface Unit.

Figure 11 shows sockets distributed around the edge of the Ncore subsystem. Table 8 shows the Cache-Coherent Interconnect Socket NIU bindings and supported protocols.

Figure 11. Subsystem and Socket Example

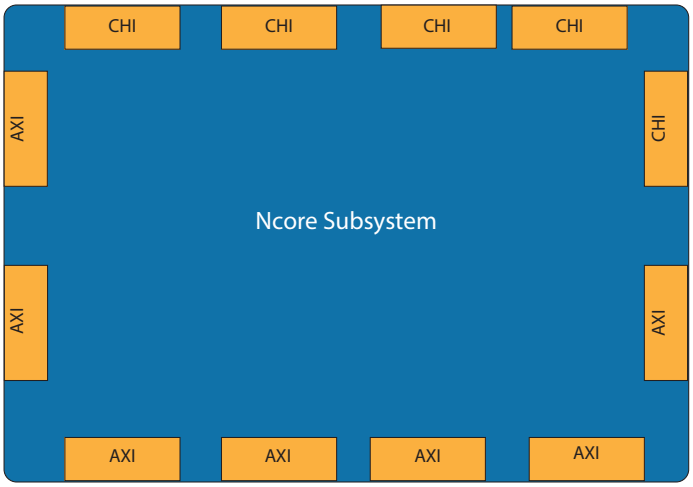


Table 8. Cache-Coherent Interconnect Socket NIU bindings and supported protocols

Interface Unit Name	Binds to socket objects and supported protocols
Coherent-Agent Interface Unit (CAIU)	ACE CHI

Table 8. Cache-Coherent Interconnect Socket NIU bindings and supported protocols

Interface Unit Name	Binds to socket objects and supported protocols
Non-Coherent Agent Interface Unit (NCAIU)	AXI 4 ACE-Lite ACE-Lite-E
Distributed Coherency Engine (DCE)	Does not bind to any socket objects.
Distributed Memory Interface (DMI)	AXI 4
Distributed IO Interface (DII)	AXI4
Distributed Virtual Memory System Engine (DVE)	Does not bind to any socket objects.

4.2 Socket Configuration Procedures

This procedure assumes you are connected to an Ncore project.

4.2.1 Common Socket Configuration Procedures

The following parameters must be configured for every socket. Note: You must be in at least the Socket Assembly phase in the Task View to perform these tasks

1. In the Project Tree, navigate to the subsystem.
2. Right-click Sockets. A menu is displayed.
3. Select Create from the menu.
4. Assign a name to the new socket. Click OK.
5. Select the down arrow in the Domain column. A list of paths to the clock subdomains paths is displayed. Scroll the list of paths. Select the path of the clock subdomain that you want to associate with the Domain parameter.
6. Select the down arrow in the Protocol column. A list of Protocol types is displayed. Select a Protocol Type to use as the Protocol parameter value.
7. Configure the values that are specific to the protocol type selected in step 6.

Common Socket Parameter Descriptions

This section describes the parameters that are common to socket configuration.

Name

The name of the coherent network interface object.

Domain

This parameter displays the paths to the clock subdomain configured for the object.

Protocol

This parameter displays a list of Protocol Types.

Function

This parameter displays the function selected for the socket.

4.2.2 Configure a CAIU Socket

Note The procedures in the following section describe how to configure socket parameter values in the Object Editor. You can also configure parameter values in the Parameter View pane.

Complete the following steps to configure the parameter values for a CAIU socket.

1. Complete the steps in section 4.2.1.
2. Enter additional parameters per your design for ACE, CHI-A, or CHI-B as described in the following steps.
3. If the protocol selected is ACE, these additional parameters should be entered:

- Width of the Ar Id: Specify the width of the Ar Id bits. Range is 1 to 20.
- Width of Aw Id: Specify the width of the Aw Id bits. Range is 1 to 20.
- AXI data bus width: Specify the data width. The data bus width options are 8, 16, 32, 64, 128, 256, 512, 1024, and 2048 bits.
- Width of Address: Specify the address width: The address width range is 12 to 64 bits.
- Width of Aw User: Specify the width of the ACE interface Aw User bits. Range is 0 to 32 bits.
- Width of Ar User: Specify the width of the ACE interface Ar User bits. Range is 0 to 32 bits.

4. If the protocol selected is CHI-A, these additional parameters should be entered:

- CHI address bus width: The address width of the CHI-A interface is fixed to 44 bits.
- Width of the request RSVDC: Specify the width of the request RSVDC/user bits. The dropdown shows a list that contains the integers 0, 4, 8, 12, 16, 24, and 32. Select an integer from the list to set the width of the request RSVDC parameter value.
- CHI data bus width: Specify the data width for the CHI-A interface. Options are 128 and 256 bits.

5. If the protocol selected is CHI-B, these additional parameters should be entered:

- Width of source Id: Width of the source ID. Set to seven and cannot be changed.
- Width of target Id: Width of the target Id. Set to seven and cannot be changed.
- Width of Node Id: Width of the Node Id of the CHI interface. Range is 7 to 11 bits.

- CHI address bus width: Specify the address width of the CHI-B interface. Range is 44 to 52 bits.
- Width of the request RSVDC: Specify the width of the request RSVDC/user bits. The dropdown shows a list that contains the integers 0, 4, 8, 12, 16, 24, 32. Select an integer from the list to set the width of the request RSVDC parameter value.
- CHI data bus width: Specify the data width for the CHI-A interface. Options are 128 and 256 bits.
- Enable poison bits: Enable poison bits for the CHI-B interface. It is set to false by default.

4.2.3 Configure an NCAIU Socket

Complete the following steps to configure the parameter values for an NCAIU socket.

1. Complete the steps in section 4.2.1.
2. If the protocol selected is ACE-Lite or ACE-Lite-E, these additional parameters should be entered:

- Click the text box in the Width of Ar ID column. Scroll the up or down arrows and set the Width of Ar ID parameter value.
- Click the text box in the Width of Aw ID column. Scroll the up or down arrows and select an integer to set the Width of AwID parameter value.
- Click the text box in the AXI address bus width column. Scroll the up or down arrows and select an integer to set the AXI address bus width parameter value.
- Click the text box in the Width of AxUser and XUser column. Scroll the up or down arrows and select an integer to set the Width of AxUser and XUser parameter value.
- Select the down arrow in the AXI data bus width column. A list displays the values. Select a value from the list to set the AXI data bus width parameter value.
- Select the down arrow in the Enable DVM column. A list of integers is displayed. Select `0` or 1 to set the Enable DVM parameter value.

3. If the protocol selected is AXI4, these additional parameters should be entered:

- Click the text box in the Width of Ar ID column. Scroll the up or down arrows and select an integer to set the Width of Ar ID parameter value.
- Click the text box in the Width of Aw ID column. Scroll the up or down arrows and select an integer to set the Width of AwID parameter value.
- Click the text box in the AXI address bus width column. Scroll the up or down arrows and select an integer to set the AXI address bus width parameter value.
- Click in the Width of AxUser and XUser column. Roll the up or down arrows to set the value of the AxUser and XUser width parameter.
- Select the down arrow in the AXI data bus width column. A list displays the values. Select an entry from the list to set the value of the AXI data bus width parameter.

4.2.4 Configure a DMI Socket

Complete the following steps to configure the parameter values for a DMI socket.

1. Complete the steps in section 4.2.1.
2. Click the text box in the Width of Ar ID column. Scroll the up or down arrows and select an integer to set the Width of Ar ID parameter value.
3. Click the text box in the Width of Aw ID column. Scroll the up or down arrows and select an integer to set the Width of AwID parameter value.
4. Click the text box in the AXI address bus width column. Scroll the up or down arrows and select an integer to set the AXI address bus width parameter value.
5. Click in the Width of AxUser and XUser column. Scroll the up or down arrows to set the value of the AxUser and XUser width parameter.
6. Select the down arrow in the AXI data bus width column. A list displays the values. Select an entry from the list to set the value of the AXI data bus width parameter.

4.2.5 Configure a DII Socket

Complete the following steps to configure a DII socket.

1. Complete the steps in section 4.2.1.
2. Click the text box in the Width of Ar ID column. Scroll the up or down arrows and select an integer to set the Width of Ar ID parameter value.
3. Click the text box in the Width of Aw ID column. Scroll the up or down arrows and select an integer to set the Width of AwID parameter value.
4. Click the text box in the AXI address bus width column. Scroll the up or down arrows and select an integer to set the AXI address bus width parameter value.
5. Click in the Width of AxUser and XUser column. Scroll the up or down arrows and select an integer to set the AxUser and XUser width parameter.
6. Select the down arrow in the AXI data bus width column. A list displays the values. Select an entry from the list to set the value of the AXI data bus width parameter value.

4.3 Protocol Parameters

When one socket is configured, the socket can be cloned (i.e., duplicated) by clicking on the socket name in the Project View, then right-mouse clicking to pop-up the hidden menu. Select "Clone".

4.4 Memory Map

The steps in this section explain how to configure the Memory Map.

Before you begin

Read the following section to become familiar with how to configure a Memory Map. The section contains examples of memory set and memory group configuration and information about interleaving.

4.5 DMI Interleaving Procedure

Sockets that are defined with the function of MEMORY can be interleaved. The Ncore units attached to sockets configured as MEMORY are the DMIs. The address regions mapped to the DMIs can be interleaved. All DMIs are assigned to one or more Memory Groups.

A maximum of four DMIs per Memory Group is permitted. The DMIs can only be placed in ascending order and allotted to Memory Groups. Even a single DMI must be assigned a memory group.

DMI Configuration Example

The following example is based on setting assignments for five DMIs. In Ncore, we can create a maximum of two Memory Sets.

MemorySet is created under Memory Map. For each MemorySet, you must declare one or more Memory Groups. Memory Set contains the entire set of MEMORY sockets distributed into different Dynamic Memory Groups.

- MemorySet `0` contains 3 groups —MemoryGroup`00`, MemoryGroup`01`, and MemoryGroup`02`:
  - DMI`0`, DMI1 in MemoryGroup`00`
  - DMI2, DMI3 in MemoryGroup`01`
  - DMI4 in MemoryGroup`02`
- MemorySet 1 contains 2 groups — MemoryGroup`10` and MemoryGroup11:
  - DMI`0`, DMI1, DMI2, DMI3 in MemoryGroup`10`
  - DMI4 in MemoryGroup11

4.5.1 Memory Interleaving Function

The Memory Interleaving Function specifies the selection bits for 2-way and 4-way interleaving schemes. When the number of DMIs in a Memory Group is two, the two-way interleaving function is used. When the number of DMIs in a MemoryGroup is four, the four-way interleaving function is used.

The primary and secondary selection bits are defined in the interleaving functions.

Referring to the example of the five DMIs:

- MemoryGroup`0` of MemorySet`0` uses a two-way interleaving function because there are only two DMIs: DMI`0` and DMI1. MemoryGroup`0` of MemorySet`0` requires 1 primary bit to specify which bits of the address are used to interleave the addresses.

- MemoryGroup0 of MemorySet1 uses four-way interleaving function because the memory group has four DMIs: DMI0, DMI1, DMI2, DMI4. MemoryGroup0 of MemorySet1 requires two primary bits to specify which bits of the address are used to interleave the addresses.

A maximum of two interleaving functions are defined for each interleaving scheme.

4.5.2 Create Memory Map

1. Select Project View.
2. Right-click Memory Map. A menu is displayed.
3. Select Create from the menu.
4. Select a name for the Memory Map and click OK.

The name of the new Memory Map is displayed below the Memory Map directory.

4.5.3 Create a Memory Set

1. In the Project Tree, expand the right arrow next to the Memory Map directory.
2. Right-click the name of the Memory Map. A list of actions is displayed.
3. Select Create > MemorySet. The Create MemorySet window is displayed.
4. Enter the name of the new MemorySet in the text field. Select OK.

The name of the new MemorySet is displayed in the Object Editor and the Parameter View. The name of the new Memory Set is displayed in the Project Tree beneath the directory of the selected Memory Map name.

To create a second MemorySet, repeat steps 1-4.

4.5.4 Create a Memory Interleave Group Set

1. Right-click the name of a MemorySet.
2. Select Create > Dynamic Memory Group. The window displays the Create Dynamic Memory Group window.
3. Enter the name of the new Dynamic Memory Group. Select OK.

The name of the new Dynamic Memory Group is displayed below the name of the selected MemorySet directory in the Project Tree.

4.5.5 Create Memory Interleaving Function

1. Expand the Memory Interleaving Functions directory.
2. Select the Memory Interleaving Functions directory.
3. Select Create > Memory Interleaving Function. The Create Memory Interleaving Function window is displayed.
4. Enter the name of the new Memory Interleaving Function.
5. Select OK. The name of the new Memory Interleaving Function is displayed in the Memory Interleaving Function directory.

6. Select the name of a Memory Interleaving Function. The Object Editor and Parameter View display the parameter names and values of the selected Memory Interleaving Function.
7. As required, configure values for:
    a. Memory Interleaving Function Identifier
    b. Primary Interleaving Bit #1
    c. Primary Interleaving Bit #2
    d. Primary Interleaving Bit #3

The Object Editor and the Parameter View display the selected Interleaving Function #1 and Interleaving Function#2 values.

4.5.6 Create Boot Region

Complete the following steps to create a boot region.

1. In the Project Tree, right-click on a Memory Map. The Create Boot Region page is displayed.
2. Enter a name for the new Boot Region in the text field.
3. Select OK. The name of the new Boot Region is added to the Boot Region directory in the Project Tree.
4. Left-click the name of the new Boot Region. The Object Editor and the Parameter View display parameter names and values for the new Boot Region.
5. Select the path to a Target Socket. Place the cursor in the field to display a list of the paths to each configured socket. Select a path from the list. For example, you might select project/chip/system/subsystem/sock_caiu0. The selected path is displayed in the Target Socket field.
6. Define the value for the Base Address. The unit for the Base Address is (kb). Enter a value for the Base Address, or select the up/down arrows to select a value. The Base Address field displays the selected value.
7. Define the Memory Size value. Enter the value or select the up/down arrows to select a value. The Size field displays the selected value.

Base Address

Specify the Boot Region base address. This address must be aligned to the size specified.

Boot Region Size

Specifies the size of the Boot Region. The minimum size is 4KB and must be a power of two.

4.5.7 Create Configuration Region

To create a Configuration Region, complete the following steps.

1. In the Project Tree, right-click on a Memory Map. The Create Configuration Region page is displayed.

2. Enter a name for the new Configuration Region in the text field.
3. Select OK. The name of the new Configuration Region is displayed in the Configuration Region directory in the Project Tree.
4. Left-click the name of the Configuration Region. The Object Editor and the Parameter View display parameter names and values for the selected Configuration Region.
5. Select the path to a Target Socket. Select the field to display a list of the paths to each configured socket. Select a path from the list. For example, you might select project/chip/system/subsystem/sock_caiu1. The selected path is displayed in the Target Socket field.
6. Define a value for the Mask parameter. Select the field and enter a value for the Mask, or select the up/down arrows to select a value. The Mask field displays the value.
7. Define a value for the Max Burst parameter. Select the field and enter a value for the Max Burst, or select the up/down arrows to select a value. The Max Burst field displays the value.
8. Enable or disable the Split Wrap parameter. To enable the parameter, select the checkbox to enable the parameter. Clear the checkbox to disable the parameter.
9. Define the value for the Base Offset. Select the field and enter a value for the Base Address, or select the up/down arrows to select a value. The Base Address field displays the selected value.
10. Define the Boundary value. Select the field and enter the value or select the up/down arrows to select a value. The Boundary field displays the selected value.

Check the console for messages. Select the Flow Navigator ribbon. Expand the System Assembly menu entry. Once complete, proceed to the Structural Design phase.

5

Structural Design (Architecture, Phase 3)

The structural design interconnect takes place in Phase 3. The Structural Design phase involves the following steps:

- Create a TransportSolution. In this section, the name of the TransportSolution is Topology.
- Open Topology and select the name of the template that was configured when Topology was created.
- Run the following automated tools on the Topology
  - Interface Inserter
  - Regular Topology Generator Wizard
  - Insert Resiliency
  - Insert Configuration Network
- Use the Mapping tool in the Flow Navigation tab to view the network designs.

5.1 Configure a Transport Solution

In this section, the TransportSolution is given the name *Topology*.

To create a TransportSolution, complete the following steps.

1. In the Project Tree, expand system > subsystem.
2. Right-click subsystem. A menu is displayed. Select Create > TransportSolution. A pane is displayed.
3. Enter Topology as the name for the TransportSolution and click OK. A directory named Topology is added to the system > subsystem> Architecture directory.
4. Select the Topology directory. The Parameter View and the Object Editor display the names of the parameters to configure Topology.
5. Depending on design requirements, configure values for the parameters listed and save the project. For example: configure values for Number of DVM command request credits, Number of DVM snoop request credits, Memory Protection, and so on.

5.2 Automation Tool Overview

Maestro provides a set of tools that automate the tasks required to complete the design. The tools are accessed through the File Menu. The automation tool names are:

- Insert Interface Units
- Regular Topology Generator Wizard
- Inset Interrupt Handling
- Insert Resiliency Handling
- Insert Configuration Networks

5.2.1 Select a Template from the TransportSolution

The template you select defines the number of control and data networks in the design. You can choose between a three-network and four-network solution.

Increasing the number of networks allow for more parallelism of communication and performance at the cost of area and power. By default, the name of control network(s) is ndn and the data network is named dn.

To select a template, complete the following steps.

1. In the Project Tree, navigate to system > subsystem > Architecture and select Topology. The Object Editor and Parameter View display all of the Topology parameters and values.
2. Scroll the list of parameters and select the down arrow in the Template menu item. A list displays two template names.
3. Select a template. The selections are:

- TwoCtrlOneDataTemplate - Two control networks and one data network.
- ThreeCtrlOneDataTemplate - Three control networks and one data network

After a template is selected, the Architecture directory is automatically updated with a new directory named Networks. The directory contains subdirectories for each control network and for the data network. A directory for Interface Units is also added to the Networks directory.

5.2.2 Run the Interface Inserter Tool

The Interface Inserter Tool automatically iterates over all of the sockets and creates one Ncore unit in front of each socket. An Ncore unit is an interface unit and an array of packetizers and depacketizers. The tool also creates one DVE, a configurable number of DCEs (the default is one), and one extra DII as an entry point into the configuration network.

To run the Interface Inserter tool, complete the following steps.

1. Select Topology.

2. Select the Interface Inserter icon in the toolbar. The Run Interface Inserter pane is displayed.

3. Verify the settings displayed on the Run Interface Inserter pane. Change the settings if required. The settings are described below.

- Work on this design: This refers to the solution created under the subsystem.
- Use this default clock: This refers to the clock subdomain created in Phase 1.
- Specify number of DCEs: You can specify the number of DCEs. Note: When there are multiple DCEs, the user will need to edit interleaving bits.

4. Select OK to run the Insert Interface tool.

5. The tool iterates over all the Sockets and creates one Ncore Unit in front of each. It also creates one DVE, a configurable number of DCEs (the default is 1), and one extra DII as an entry point into the configuration network.

The console displays messages about the progress of the Interface Inserter tool. In the Project Tree, the Networks directory is updated with information about the interface units.

5.2.3 Run the Regular Topology Generator Wizard Tool

The Regular Topology Generator Wizard automatically creates an interconnect between the endpoints of each network (see "Topologies").

Depending on the template you selected, you must run the Regular Topology Generator Wizard tool at least two times to generate 2 control networks and then run the Regular Topology Network to generate 1 data network. For example, if the selected is configured for two control networks and one data network, you would run the tool three times.

Note To manually configure the network connections, do not run the Regular Topology Generator Wizard. Open the Topology Editor. The Topology Editor has a set of tools that you can use to create switches and define routes between endpoints. The tools allow to take actions such as inserting adapters and Pipe Adapters. Pipe Adapters (for timing) must be inserted manually. Width adapters, rate adapters, and clock adapters, which bridge different widths or frequencies of network elements, are automatically inserted if you run the Regular Topology Generator Wizard.

To run the Regular Topology Generator Wizard, complete the following steps.

1. Select the Transport Solution.
2. Select the Regular Topology Generator Wizard icon. The Regular Topology Generator Wizard pane is displayed.
3. Select the down arrow in the Generator Type: list.
4. Choose a generator type:

- mesh
- torus
- crossbar
- ring
- butterfly

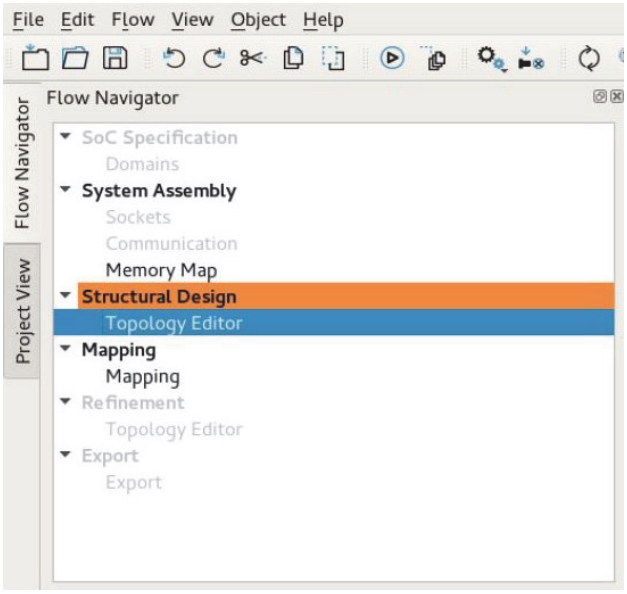
5. Select the down arrow in the Network: list.
6. Select a network name.
7. Verify that the clock subdomain field displays the correct name of the clock subdomain.
8. Select Next.
9. Configure the network settings and select Next. A new pane displays a summary of the configuration settings.
10. Review the settings. If the settings are correct, select Finish to run the Regular Topology Generator Wizard. Select Back to change any of the settings listed in the summary.

The console displays messages about the progress of the Regular Topology Generator Wizard. In the Project Tree, the Networks directory is updated.

5.2.4 Topology Editor — View the Structural Design Architecture

1. Select the Flow Navigator Tree.
2. In the left pane, select Structural Design > Topology Editor.
3. The Topology Editor displays the network design and displays what has been created.

Figure 12. Structural View > Topology Editor



Retrieving a closed topology editor

While working on the topology editors, if you close any of the topology editors for a network, you can retrieve it back using the following steps.

1. Select Window -> Views -> Topology Editor.
2. This will bring back all the closed topology editors.

5.2.5 Run the Insert Interrupt Handling Tool

The Insert Interrupt tool automatically creates and connects all interrupt pins. To run the Insert Interrupt Handling tool, complete the following steps.

1. Select Topology. From the toolbar, select the Insert Interrupt Handling icon. The Run Generator pane is displayed.
2. Verify the following settings displayed on the Run Generator pane. Change the settings if required.

- Work on this design: The path points to the topology folder.
- Use this default clock: The path points to the clock subdomain.

3. Select OK to run the Insert Interrupt tool.
4. The console displays messages about the progress of the Insert Interrupt tool.

5.2.6 Run the Insert Resiliency Tool

Running the Insert Resiliency tool is only required if the Resiliency parameter in Topology is enabled.

1. Select the Project Tree.

2. Select Topology.
3. Scroll the Parameter View pane to see if the Resiliency checkbox is enabled. If the Resiliency parameter is enabled, complete the following steps.
4. From the toolbar, select the Insert Resiliency icon. The Run Generator pane is displayed.
5. Verify the following settings displayed on the Run Generator pane. Change the settings if required.
   a. Work on this design: The path points to the topology folder.
   b. Use this default clock: The path points to the clock subdomain.
6. Select OK to run the Insert Resiliency tool.

The console displays messages about the progress of the Insert Resiliency tool.

5.2.7 Run the Insert Configuration Network Tool

The Insert Configuration Network Tool locates all of the units that have a programmable APB that connects all programmable elements of the design, allowing for at-chip-runtime configuration and querying of all elements.

The tool automatically inserts the CSR network to connect all configuration registers. After you run the Configuration Network tool, the Topology Editor is updated with two tabs that you can select to view the output of the Configuration Network tool.

1. From the toolbar, select the Insert Configuration Network icon. The Run Generator pane is displayed.
2. Verify the following settings displayed on the Run Generator pane. Change the settings if required.

- Work on this design: The path points to the Topology folder.
- Use this default clock: The path points to the clock subdomain.

3. Select OK to run the Insert Configuration Network tool.
4. Select the Navigation flow tab. Scroll to Structural Design > Topology Editor. Verify the presence of one tab named Topology Editor - csr.request.nw and one tab named Topology Editor - csr.response.new.

To view the output of the Configuration Network tool, select the Flow Navigator Tree and select Refinement > Topology Editor. In the Topology Editor, select the csr_request_nw tab or select the csr_response.nw tab.

5.3 Topologies

The following topologies are supported:

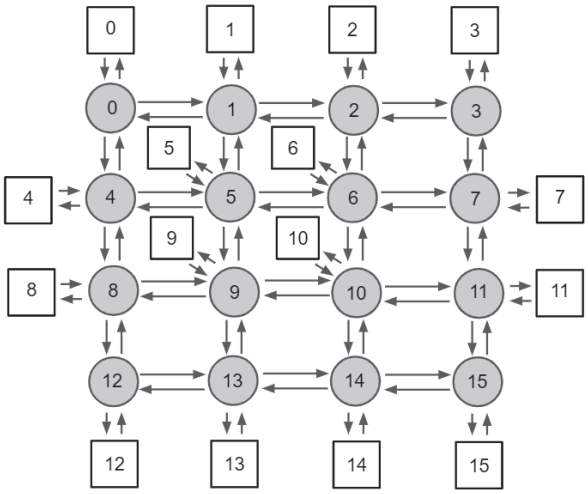
- “Mesh Topology”
- “Torus Topology”
- “Ring Topology”

- "DoubleC Topology"
- "Full Crossbar Topology"
- "Butterfly Topology"

5.3.1 Mesh Topology

In this topology, the switches are arranged as a mesh and the routes are generated based on the shortest path from the source to the destination. The switches with 1 input and 1 output port are deleted after the generation of the routes and replaced with a wire in-place.

The size of the mesh can be defined by the parameters 'meshx' and 'meshy'. The bandwidth for routing the data in the data network can be specified by the parameter 'datawidth'. The frequency for routing is defined by the clock subdomain passed in to the parameter 'clock'.

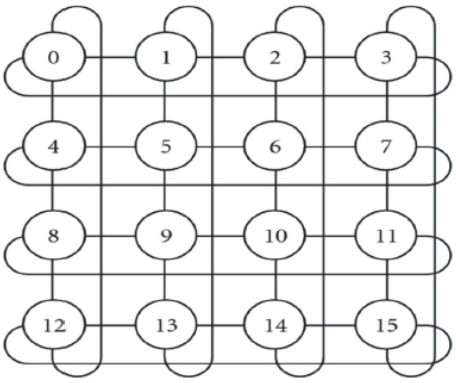


Circles indicate switches, squares indicate the Ncore units

Prior to generating routes, each Ncore unit needs to set a position on how the Ncore unit is attached to the mesh.

5.3.2 Torus Topology

The parameters of torus are similar to the parameters of mesh. The size of the torus can be defined by the parameters ‘meshx’ and ‘meshy’. The frequency for routing is defined by the clock subdomain passed in to the parameter ‘clock’.

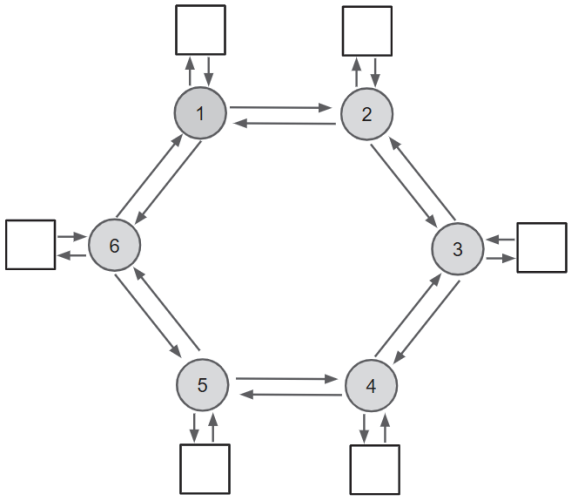


Circles indicate switches

Prior to creating the torus topology, each Ncore unit needs to assign a position as in the mesh topology.

5.3.3 Ring Topology

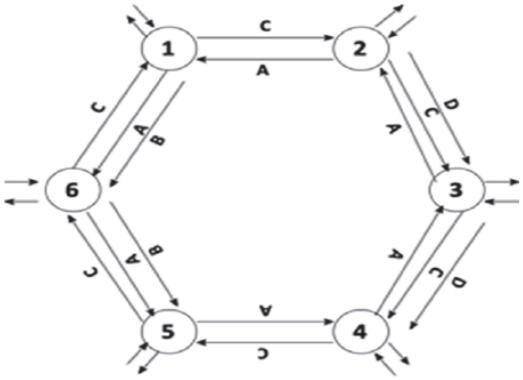
Here the switches are arranged as a form a ring. Deadlock can be resolved by the use of Virtual Channels. The user will need to set_node_position for each unit as in the x value to specify the order of the ring. The y value is set to `0`.



Circles indicate switches, squares indicate the Ncore units

5.3.4 DoubleC Topology

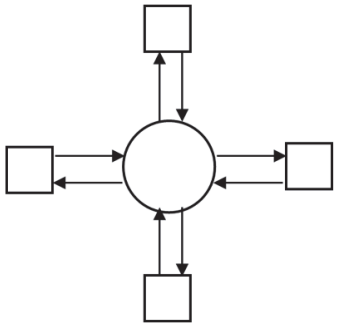
Here the switches are arranged as to form a ring. Deadlock is avoided by having parallel routes. This topology also enforces that the y coordinate is always set to `0`.



Circles indicate switches

5.3.5 Full Crossbar Topology

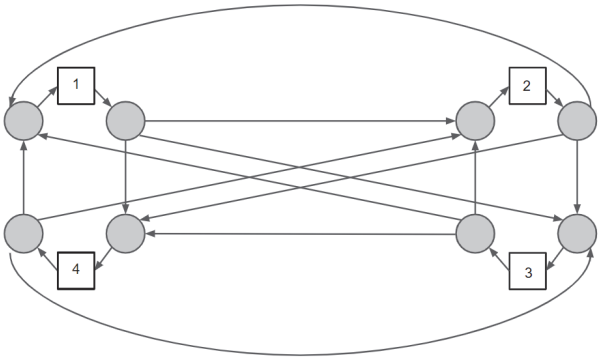
This topology consists of a single switch. All ports of all units (of the same network) will be connected to this switch.



Circles indicate switches, squares indicate the Ncore units

5.3.6 Butterfly Topology

This topology consists of each unit connected to its own switch. The switch is connected to the other switches on the network. Therefore, each connection between each unit is connected by two switches.

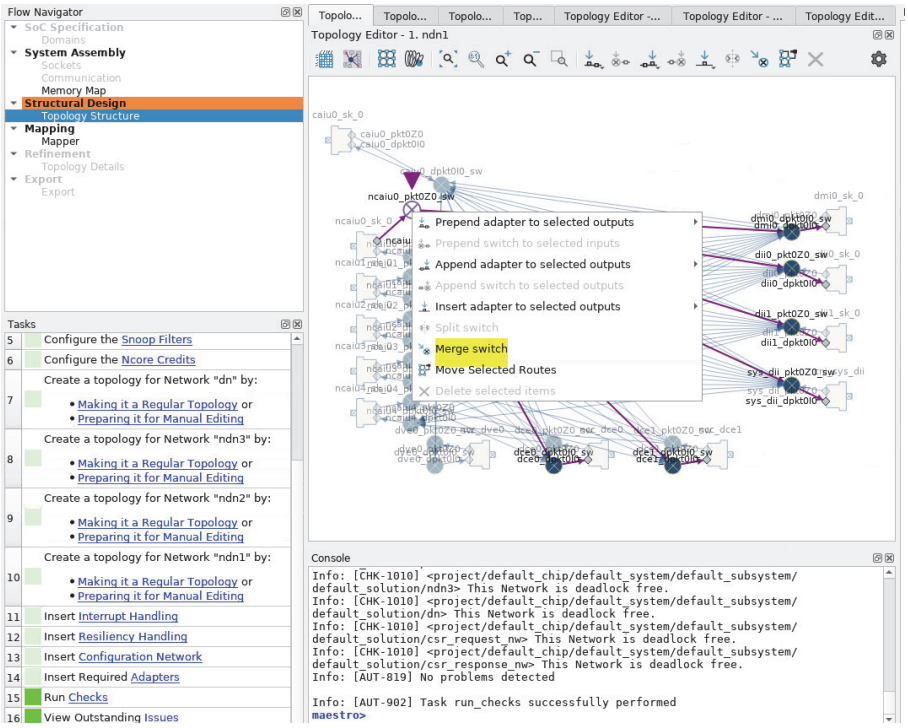


Circles indicate switches, squares indicate the Ncore units

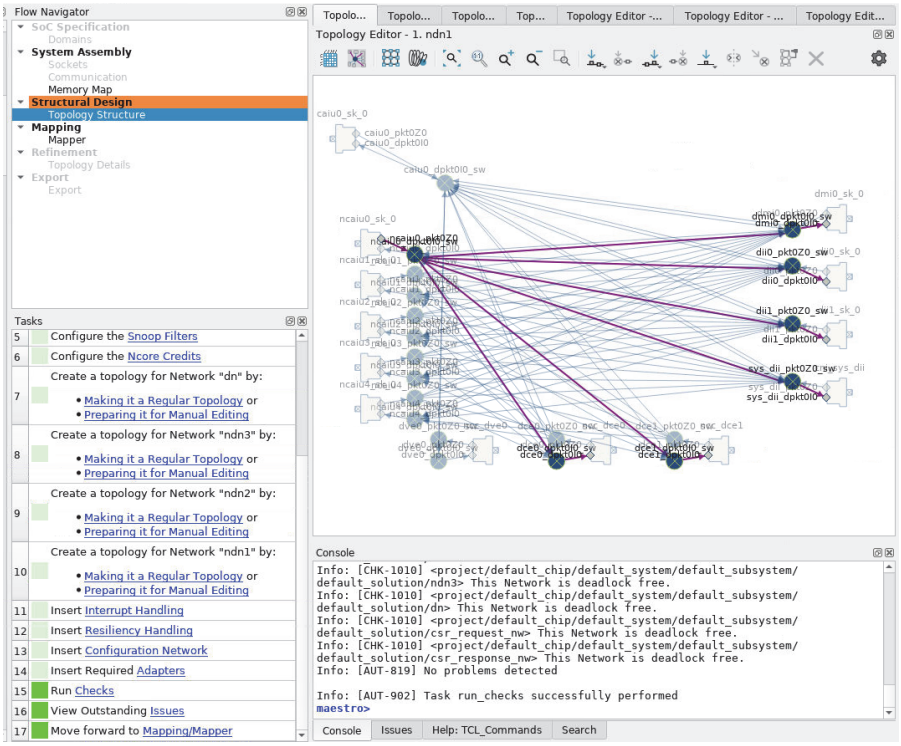
5.4 Merging Two Switches

Use these steps to merge two switches into one switch.

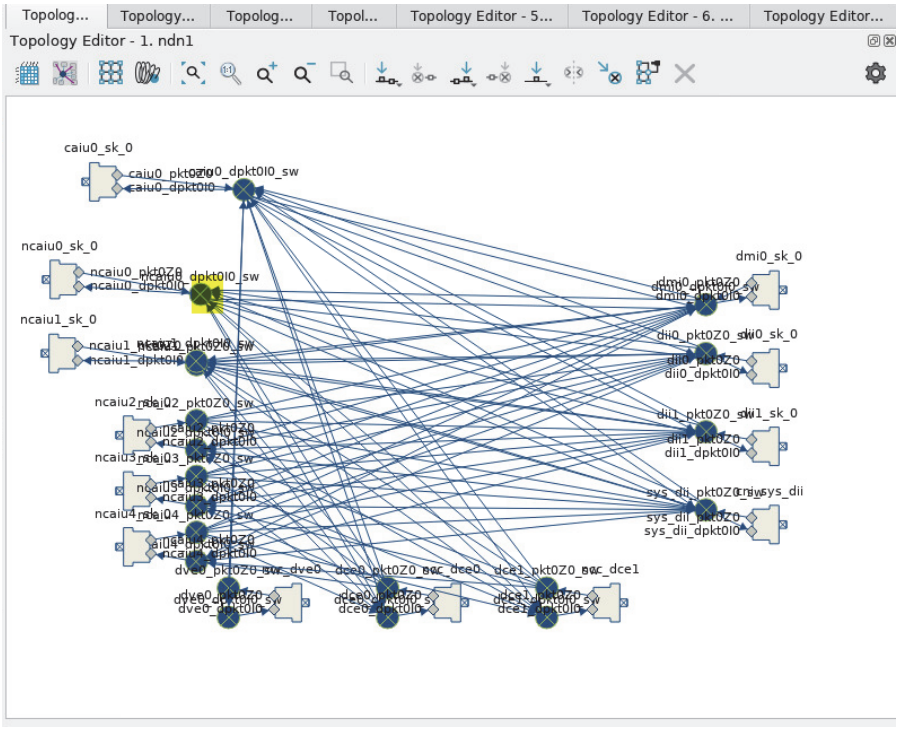
1. In the Topology editor, left-click on a switch and use Ctrl + left-click on all the switches you desire to merge.
2. You can then either use the toolbar Merge switch command, or right-click and select the Merge switch.



3. Drag it and place it on top of the switch you want to merge with.



4. The two switches merged into one switch.



6 Mapping (Phase 4)

In this phase of the design flow, Maestro associates specific RTL instances and detailed parameters to the design completed in the Structural Design (Phase 3). The new associated parameters and instances become accessible simply by selecting the Refinement > Topology Editor in the Flow Navigator and clicking on the same units that had been previously populated.

The RTL instances are configured based on all the parameters and structural properties specified in Phases 1 through 3. For example, the routing tables are populated based on information about units, address maps, connectivity, and routes.

6.1 Run the Mapping Tool

Mapping is done automatically after you clicked “Move forward to Mapping/ Mapper” from the structural design phase. You should see messages scrolling in the window as the mapping process executes automatically.

1. In Tasks View, click “Run Checks”.
2. If the console displays “No deadlocks found”, the process completed successfully.
3. Make sure there are no errors (View Outstanding Issues).
4. To view the design after the Mapping phase is complete, select the Flow Navigator Tree and select Refinement > Topology Editor.

7

Architectural Refinement (Phase 5)

This phase provides a detailed parameterization of the RTL units for fine tuning of feature and performance area tradeoffs. All of the objects you can select in the Project Tree or in the Topology Editor now have a full set of hardware parameters that can be edited.

7.1 Refine the Design Architecture

To view a graphical rendering of each network in the design, complete the following steps.

1. Select the Flow Navigator pane.
2. Expand Refinement > Topology Editor. The Topology Editor in the main window is updated with a series of tabs. Each tab shows a network diagram that shows the interconnections. The naming convention of the tabs is described in the following example:

Example: In a network diagram that is configured with three control networks, one data network, and two configuration networks, the Topology Editor would display seven tabs:

- Topology Editor - 1. ndn1
- Topology editor - 2. ndn2
- Topology Editor - 3. ndn3
- Topology Editor - 4. dn
- Topology Editor - 5. csr_request_nw
- Topology Editor - 6. csr_response.nw
- Topology Editor - 7. All Networks

Select any tab to show the network diagram associated with the tab.

7.2 Topology Editor Tools and Controls

The Topology Editor provides tools and controls that you can use to edit the network diagrams.

7.2.1 Topology Editor

Use the Topology View Editor to select an object in the network diagram. After you select the object, the Parameter View displays the parameter names and values configured for the selected object. See “Topology Editor Toolbar” on page 23 for details on the toolbar.

7.2.2 Parameter View

The Parameter View displays the parameter names and values defined for a selected object.

You can change the values for the selected object in the Parameter View.

7.2.3 Filters Editor

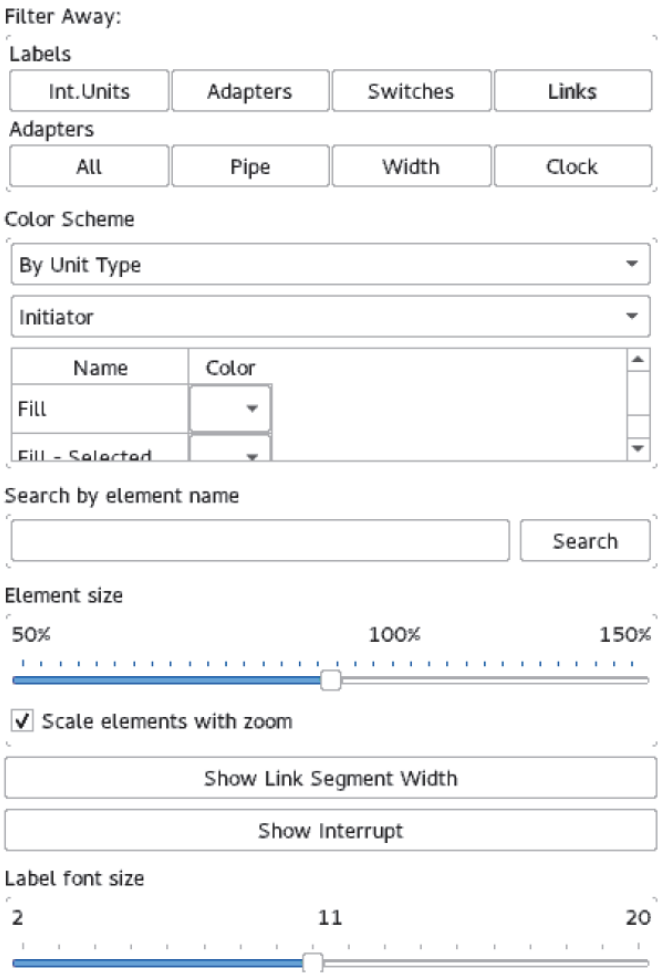
You can use the Filters editor to control the appearance and presentation of objects in a network topology editor.

Important: Changes made in the Filters editor are applied immediately to all network diagrams.

Example 1: If you select the All button, then all adapters are hidden in all network diagrams.

Example 2: If you select the Show Interrupt button, all network diagrams are updated to display the Interrupt view.

Figure 13. Filters editor example



Labels

If a button in the Labels section is greyed-out, then the associated label is hidden in the network diagram. If a button is active, then the label is displayed in the network diagram.

The network diagram is updated after you select a button in the labels section.

Int. Units — Select to show or hide the Network Interface labels in the Topology Editor network diagram.

Adapters — Select to show or hide the adapters in the Topology Editor network diagram.

Links — Select to show or hide the adapter labels in the Topology Editor network diagram.

Switches — Select to show or hide the Switch labels in the network diagram

Adapters

This section controls whether the following three elements are displayed. Select the All button to show or hide the elements. The network diagram is updated after you select a button in the network diagram.

All — Select to show or hide all Adapters.

Pipe — Select to show or hide Pipe Adapters.

Width — Select to show or hide Width Adapters

Clock — Select to show or hide Clock Adapters

Color Scheme

The steps in this section explain how to control the color and other appearance parameters when an icon displayed in a network design. Use the pulldowns to make different selections other than 'By Unit Type' ('By Clock Doman') and 'Initiator' ('Target', 'MulticastStation', Switch, etc.).

The following list describes the parameter names.

- Fill: Sets the fill color displayed for an object.
- Fill - Selected: Sets the fill color displayed for a selected object.
- Pen: Sets the color of the border of an object.
- Pen - Highlighted: Sets the color of the border of an object after the object is highlighted.
- Pen - Selected: Sets the color of the border of an object after the object is selected.

In the Style section, select an element from the list and configure the color style settings for the selected element.

Search by element name

Enter the name of an element and select Search. For example, enter 'dce' and all of the dce elements are highlighted in the network diagram.

Element size

Move the slider left or right to determine the size of the elements in the network diagram.

Show Link Segment Width

Toggle the button to show or hide the link segment in the network diagram.

Show Interrupt

Toggle the button to show or hide the Interrupt view of the network diagram.

Label font size

Move the slider to select the font size of the labels shown in the network diagram.

7.2.4 Connectivity Table and Routing Table

Each blue square in the connectivity table indicates an interconnection between two points.

To show the route of an interconnect, select a blue square in the connectivity table. A colored line shows both ends of the interconnect selected in the connectivity table.

You can also click a name on the side to highlight an entire row or column of connections at a time while inside the connectivity table.

8 Export Design (Phase 6)

In this phase of the design flow, Maestro creates the RTL and all collaterals for the design.

8.1 Export a Design

To export a design, complete the following steps.

1. Review the console to ensure that no errors are displayed.
2. Fix any errors.
3. From the toolbar, select the Generate RTL icon. The Export Design pane is displayed.
4. The default title of the Export Design is Export data set.
5. The directory names, subdirectory names and file names contained in the Export data set are described in the following list:

- RTL
- Exchange formats
  - IP-XACT
- Implementation scripts
  - Synthesis scripts
- Verification environment
  - UVM Workbench
- SystemC models >Timing Modes
  - SystemC AT/LT modes
  - Cycle Accurate

6. If required, select Browse to change the default directory in which the Export Design package is placed. The default output path is ./export.
7. Select OK.

9 Maestro Demonstration Design

9.1 Introduction

The design in this chapter is similar to the design used in the *Tcl User Guide*. It may be instructive for some users to compare how the design is created using Tcl commands and in the Maestro GUI, as shown here.

The workflow mirrors Maestro and is divided into six phases, each containing a set of tasks. It is recommended that you save your design after completion of each phase.

- "Phase 1 - SoC (Chip) Specification"
- "Phase 2 - System Assembly"
- "Phase 3 - Structural Design (Architecture)"
- "Phase 4 - Mapping"
- "Phase 5 - Refinement"
- "Phase 6 - Export"

The design specifications are described in the following section.

9.2 Design Specifications

Table 9. Sockets

Socket	Interface Unit
CHI-B	Caiu0
CHI-B	Caiu1
AXI4	Ncaiu0 (with proxyCache)
AXI4	Ncaiu1 (with ProxyCache)
ACE-Lite-E	Ncaiu2 (no proxyCache)
AXI4	Dmi0
AXI4	Dmi1
AXI4	Dmi2
AXI4	Dii0

Table 10. Directories

Directory	Number of SnoopFilters
Dce0	2
Dce1	2

Table 11. DMI Interleaving

DMI Interleaving	Naming
MemorySets	Ms0
	Ms1
Memory Groups	Ms0 : mg0, mg1
	Ms1 : mg0, mg1, mg2

Table 12. Regions

Region	Number	Base	Size
General Purpose Memory Regions	6		
Boot region	1	0x0	0x4000 (16384)
Configuration region	1	0x2e6e4000 (778977280)	0x400 (1024)

Table 13. Network Topology

Network	Topology
Data Network (dn)	butterfly
Non-data Network (ndn1)	manual
Non-data Network (ndn2)	butterfly

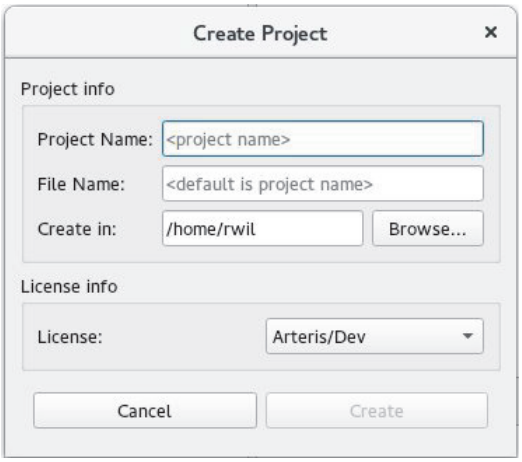
9.3 Phase 1 - SoC (Chip) Specification

Tasks for this phase:

- "Create Project" — Also see "Create a Chip"
- "Create Clock and Power Regions/Domains" — Also see "Create a Power Domain" and "Create a Clock Domain".
- "Phase 1 Checks" — Ensure the design is working before moving to next phase.

9.3.1 Create Project

1. Launch Maestro.
2. Enable Project / Tasks View (File ->Preferences -> Show Project View / Show Tasks View checkboxes).
3. Navigate to File -> New project.



4. Enter the Project Name and File Name for your project.
5. Use the Browse button to select the location for your project.
6. In the license dropdown, select "Arteris/Dev", or select your license key, if available.
7. Click Create.
8. Click the Flow Navigator icon (toggle) in the toolbar to display the ribbon which shows the current design phase, which is "SoC Specification".

9. In Tasks View, you will see a set of tasks for this phase. Click "Initialize an Ncore design".

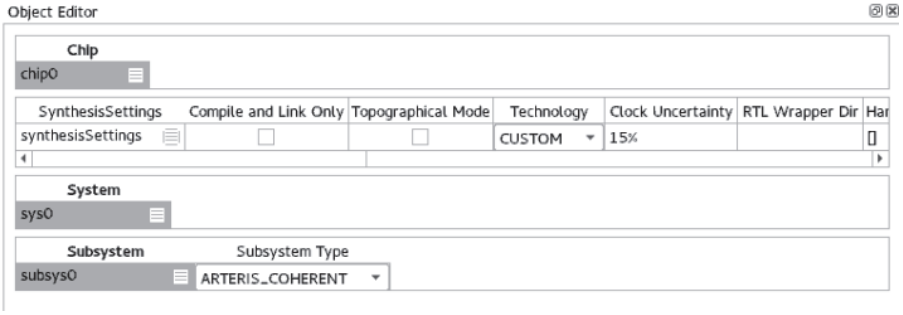
   The Project Tree will be populated with a default chip, system, and subsystem.

10. Click "default_chip" and in the Object Editor, change the following names (you can also double-click the name in the Project Tree and change it there as well):

a. default_chip -> chip0

b. default_system -> sys0

c. default_subsystem -> subsys0



9.3.2 Create Clock and Power Regions/Domains

11. In Tasks View, click “Create Power/Clock Partitioning”. This will add a default clock region, clock domain, clock sub domain, power region, and power domain.
12. Select Project View, and click “chip0”.
13. In the Object Editor, change the following names:

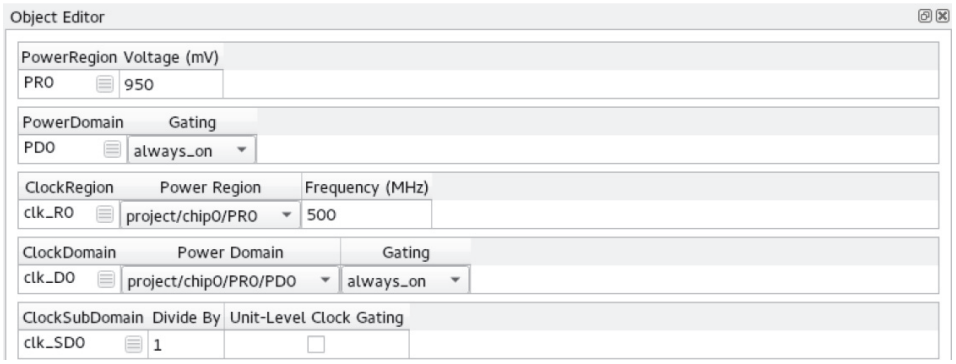
    d. default_power_region -> PR0

    e. default_power_domain -> PD0

    f. default_clock_region -> clk_R0

    g. default_clock_domain -> clk_D0

    h. default_clock_sub_domain -> clk_SD0



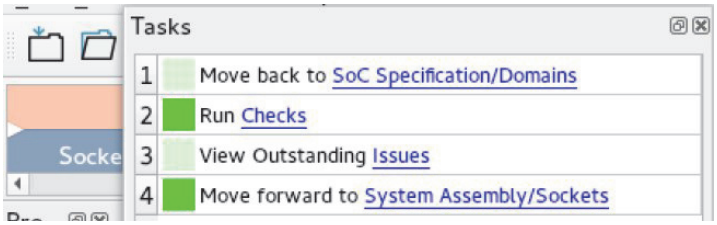
14. In the Object Editor:

    a. Go to Power Region ->PR0. Modify Voltage (mV) on the PowerRegion as: 950mV -> 1000mV

    b. Go to Clock ->clk_R0. Modify Frequency (MHz) in the Parameter View as: 500MHz -> 1600MHz

9.3.3 Phase 1 Checks

15. In Tasks View, click “Run Checks”. You should see an information message in the console saying “"...Task run_checks successfully performed".
16. Make sure there are no errors (View Outstanding Issues).



17. If there are no outstanding issues, proceed to “System Assembly/Sockets”.

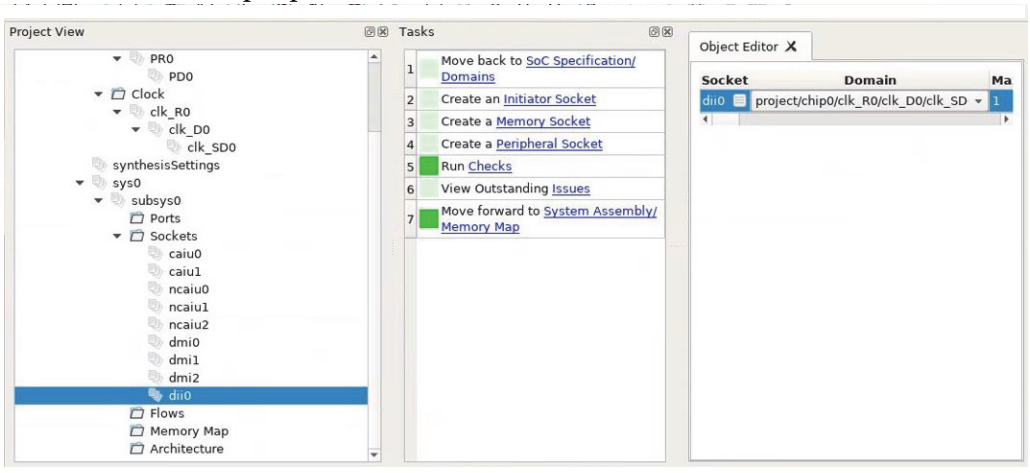
9.4 Phase 2 - System Assembly

Tasks for this phase:

- "Create Sockets" — Also see "System Assembly Socket Overview"
- "Create Memory Map" — Also see "Memory Map"
- "Phase 2 Checks" — Ensure the design is working before moving to next phase.

9.4.1 Create Sockets

1. For this design, we want: 5 INITIATOR sockets, 3 MEMORY sockets, and 1 PERIPHERAL socket. Go to Task view and do the following:
   a. Click "Create an Initiator Socket" — 5 times
   b. Click "Create a Memory Socket" —3 times
   c. Click "Create a Peripheral Socket" — 1 time
2. In the Project Tree, click "Sockets".
3. If you haven't already done so, open Parameter View (File ->Preferences -> check Show Param Editor).
4. In the Object Editor, change the following socket names:
   a. default_initiator_socket -> caiu0
   b. default_initiator_socket_0 -> caiu1
   c. default_initiator_socket_1 -> ncaiu0
   d. default_initiator_socket_2 -> ncaiu1
   e. default_initiator_socket_3 -> ncaiu2
   f. default_memory_socket -> dmi0
   g. default_memory_socket_0 -> dmi1
   h. default_memory_socket_1 -> dmi2
   i. default_peripheral_socket -> dii0



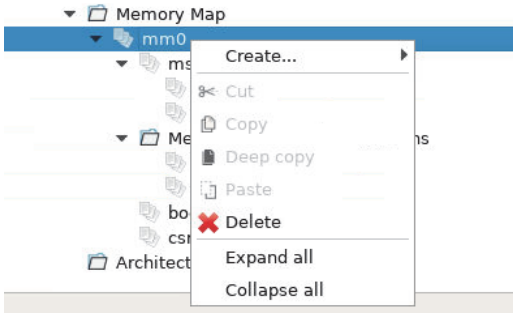
5. In the Object Editor, review/set the following parameters for each socket you just created and renamed (9 total).

Parameter	Protocol	Function	CHI/AXI address bus width	CHI/AXI data bus width
caiu0	CHI-B	INITIATOR	48	128
caiu1	CHI-B	INITIATOR	48	128
ncaiu0	AXI4	INITIATOR	48	128
ncaiu1	AXI4	INITIATOR	48	128
ncaiu2	ACE-Lite-E	INITIATOR	48	128
dmi0	AXI4	MEMORY	48	128
dmi1	AXI4	MEMORY	48	128
dmi2	AXI4	MEMORY	48	128
dii0	AXI4	PERIPHERAL	48	128

6. In Tasks View, click “Run Checks”. Make sure there are no errors or outstanding issues.
7. Click “Move forward to System Assembly/Memory Map”.

9.4.2 Create Memory Map

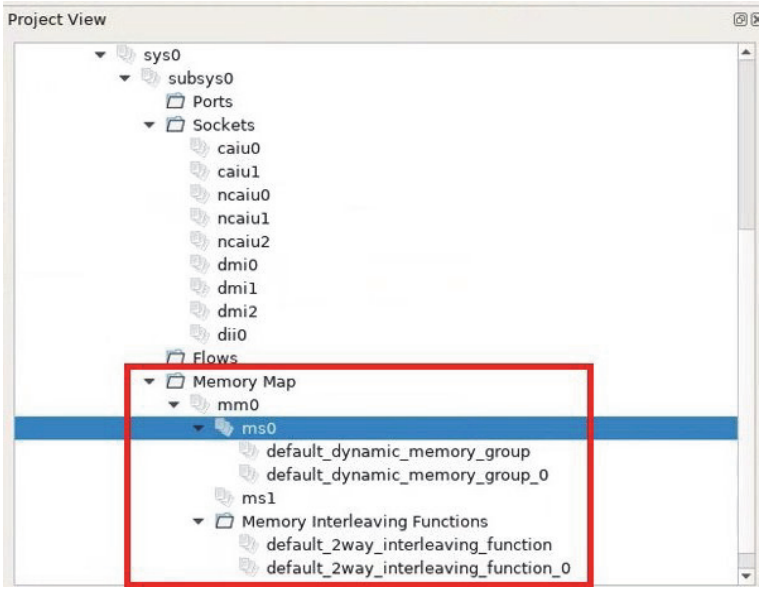
8. In Tasks View, click “Create a Memory Map”.
   a. In the Object Editor, change the following: default_memory_map -> mm0
9. In Tasks View, click “Create a Memory Set”.
   a. In the Object Editor, change the following: default_memory_set -> ms0
10. In the Project Tree, create another memory set by right-clicking the mm0 under MemoryMap.



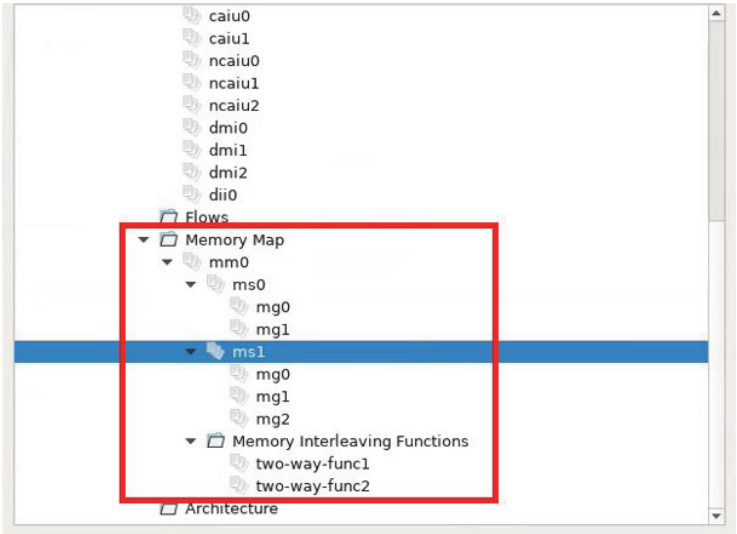
11. Click Create ->MemorySet and fill in the Input ‘MemorySet’ name with “ms1”.
12. In Tasks, click ‘Partition MemorySet “ms0“ into Dynamic Memory Groups and interleaving functions’.

    This creates two groups under mso: default_dynamic_memory_group and default_dynamic_memory_group_0. It also creates two Memory Interleaving

functions: default_2way_interleaving_function and default_2way_interleaving_function_0.



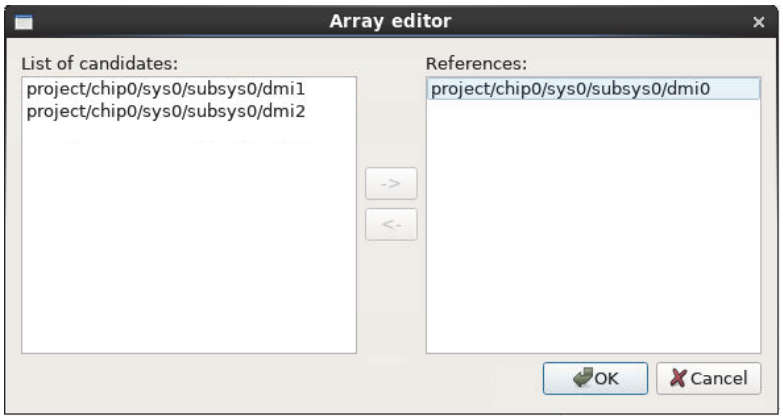
13. In Object Editor, rename these default settings as follows:
    a. default_dynamic_memory_group -> mg0
    b. default_dynamic_memory_group_0 -> mg1
    c. default_2way_interleaving_function -> two_way_func1
    d. default_2way_interleaving_function_0 -> two_way_func2
14. Since the memory groups for ms1 need to be created, do three right-clicks on ms1 in the Project Tree and create three DynamicMemoryGroups under it. Name them "mg0", "mg1", and "mg2".
15. In Project View, the Memory Map should look like this.



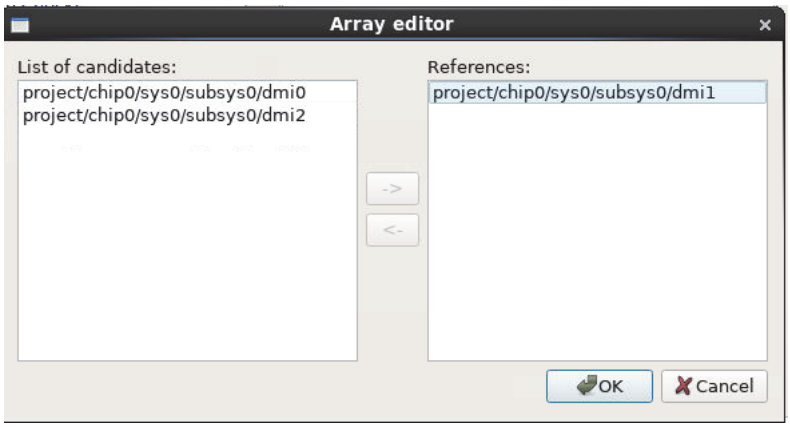
16. Click ms1. Looking at the Object Editor, notice that the Target Sockets are empty. We need to set them.

17. For memorySet **ms1**:

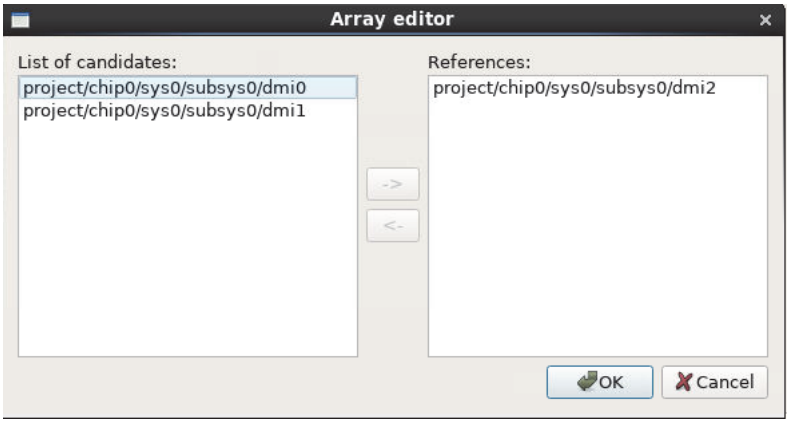
    a. Fill in the Target sockets mg0 -> dmi0.



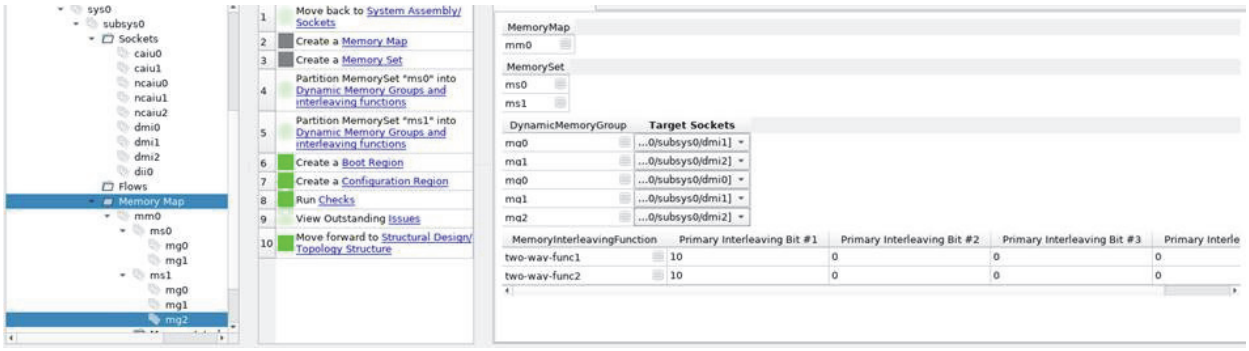
    b. Fill in the Target sockets for mg1 -> dmi1.



    c. Fill in the Target sockets for mg2 -> dmi2.



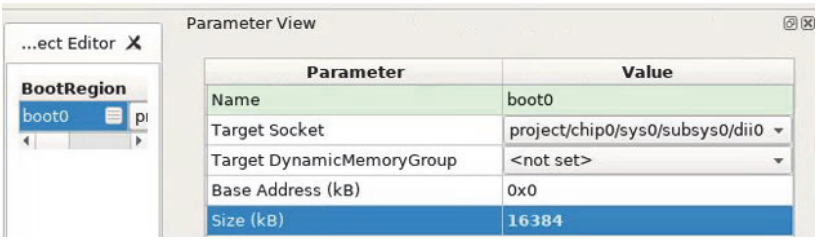
18. Your Memory Map should be similar to this.



19. In Tasks view, click “Create a Boot Region”.

20. Rename and set the boot region as follows:

    default_boot_region -> boot0
    Set Base Address -> 0x0, Size -> 16384



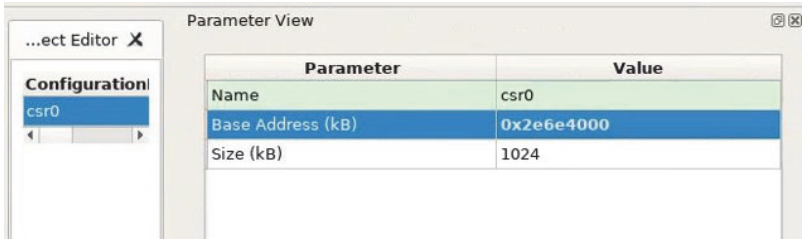
21. Verify that the Target Sockets property of the boot region boot0 is set to dii0:

    Target Sockets -> project/chip0/sys0/subsys0/dii0

22. In Tasks view, click “Create a Configuration Region”.

23. Rename and set the configuration region as follows:

    default_configuration_region -> csr0
    Set Base Address -> 0x2e800000, Size -> 1024



9.4.3 Phase 2 Checks

24. In Tasks View, click “Run Checks”. Make sure there are no errors (View Outstanding Issues).

25. Proceed to Phase 3 by clicking “Move forward to Structural Design/Topology Structure”.

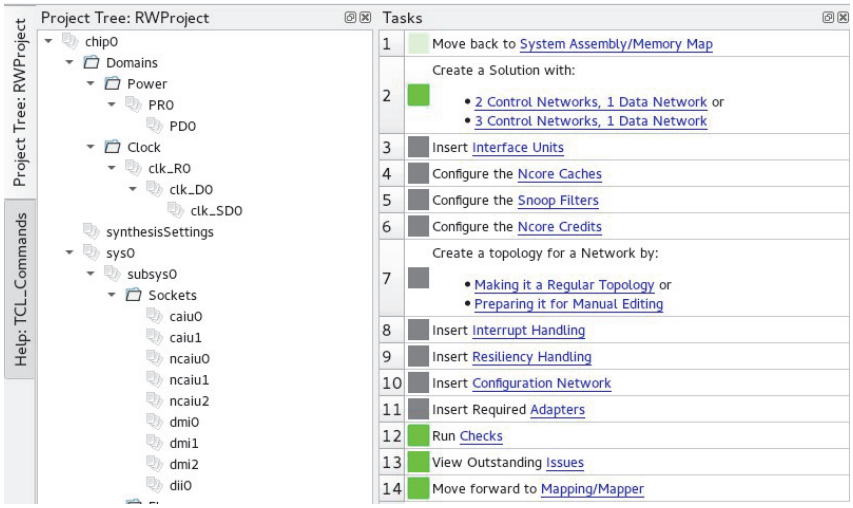
9.5 Phase 3 - Structural Design (Architecture)

Tasks for this phase:

- "Create Topology" — Also see "Structural Design (Architecture, Phase 3)"
- "Insert Interface Units" — Also see "Run the Interface Inserter Tool"
- "Configure Ncore Caches"
- "Configure Snoop Filters"
- "Configure Ncore Credits"
- "Create Topology" — Also see "Run the Regular Topology Generator Wizard Tool"
- "Insert Interrupt Handling" — Also see "Run the Insert Interrupt Handling Tool"
- "Insert Configuration Network" — Also see "Run the Insert Configuration Network Tool"
- "Set Parameters"
- "Phase 3 Checks"— Ensure the design is working before moving to next phase.

9.5.1 Create Topology

1. For this phase, you should see a screen similar to the following.



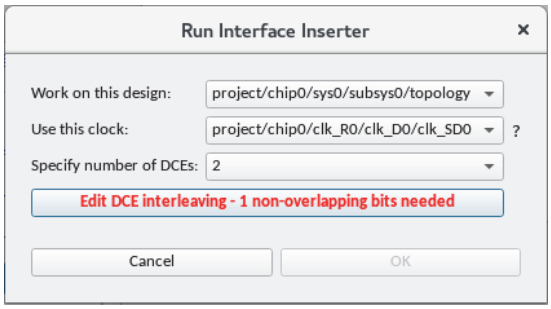
2. In Tasks View, click "Create a Solution with 2 Control Networks, 1 Data Network".
3. Under the Project Tree, change the name:

   default_solution -> topology

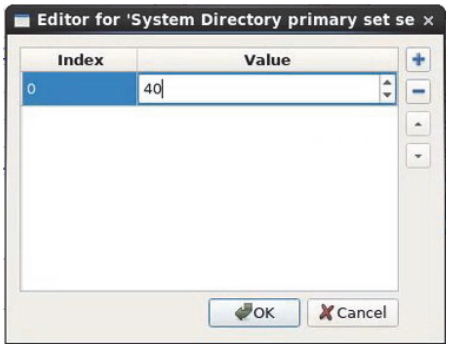
Note: If the GUI doesn't update, click the Redraw icon in the Toolbar.

9.5.2 Insert Interface Units

4. Click “Insert Interface Units”.



5. Change the number of DCEs to 2.
6. Click the “Edit DCE interleaving...” warning button.
7. Use the + button and set the value for "System Directory primary set select bits" to `40` for Index `0`, and click OK.

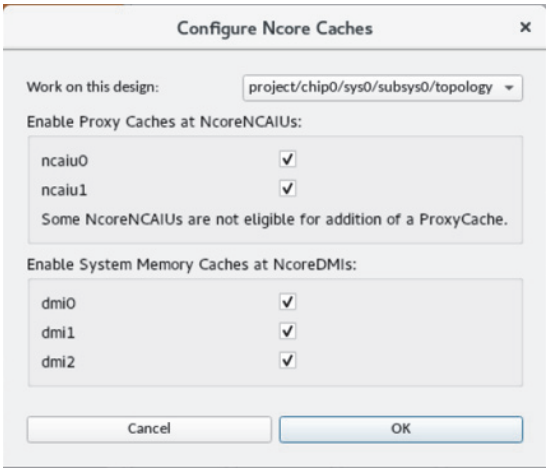


8. Click OK. An automation routine runs and iterates over all the Sockets defined in “Create Sockets” and creates one Ncore Unit in front of each. It also creates one DVE, a configurable number of DCEs (which should be specified in the column), and one extra DII as an entry point into the configuration network. Note: If you have more than one DCE, you need to have an interleaving scheme.

9.5.3 Configure Ncore Caches

9. “Click Configure Ncore Caches.”

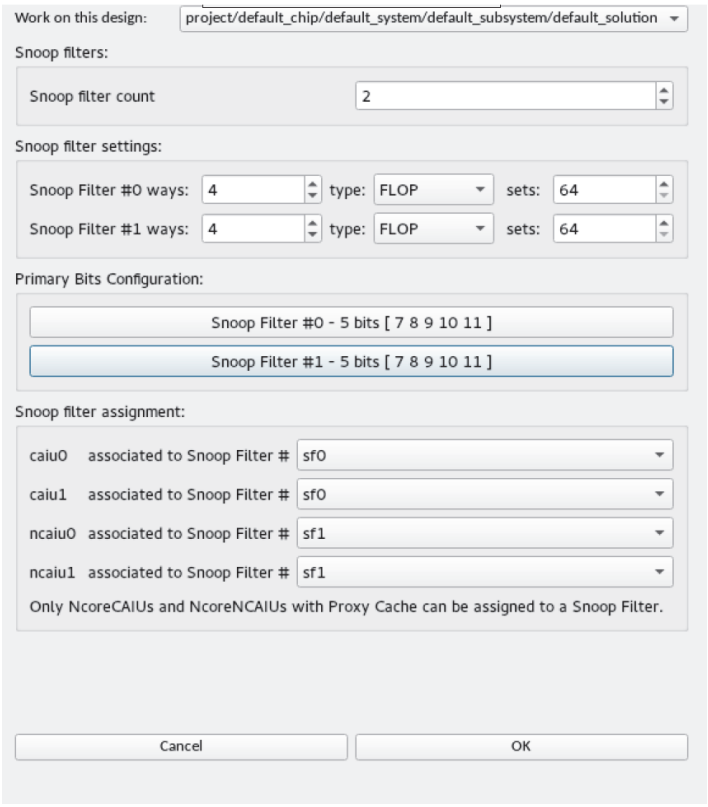
10. Select the units which have caches (check all), and click OK.



9.5.4 Configure Snoop Filters

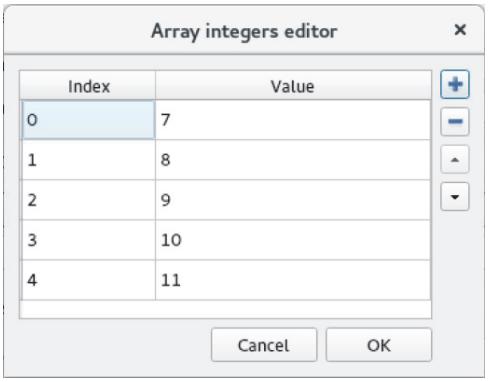
11. Click “Configure the Snoop Filters.”

12. Enter the number of Snoop Filters (2), configure the Primary Bits, and assign Snoop Filters as shown here. Click OK. Note that all Ncore3 designs must have a Snoop Filter.



13. To configure Primary Bits for the two snoop filters:

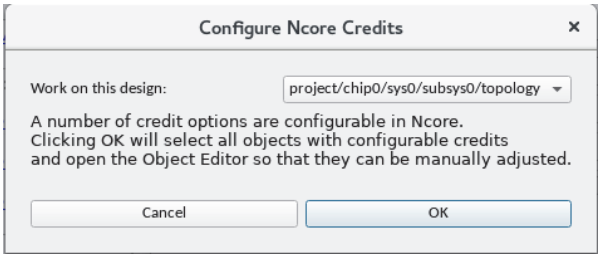
a. Click the “Snoop Filter `#0` ...” button, and set the values as follows, then click OK..



b. Click the “Snoop Filter `#1` ...” button, and set the same values as above.

9.5.5 Configure Ncore Credits

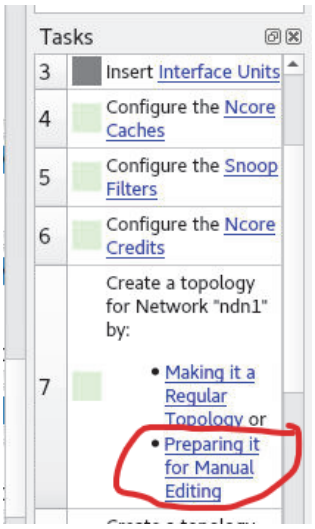
14. Click “Configure the Ncore Credits.”
15. Click OK to see the credits for the design. On the left side, all the devices with credits will be highlighted. You can update the credits for performance, or leave as default.



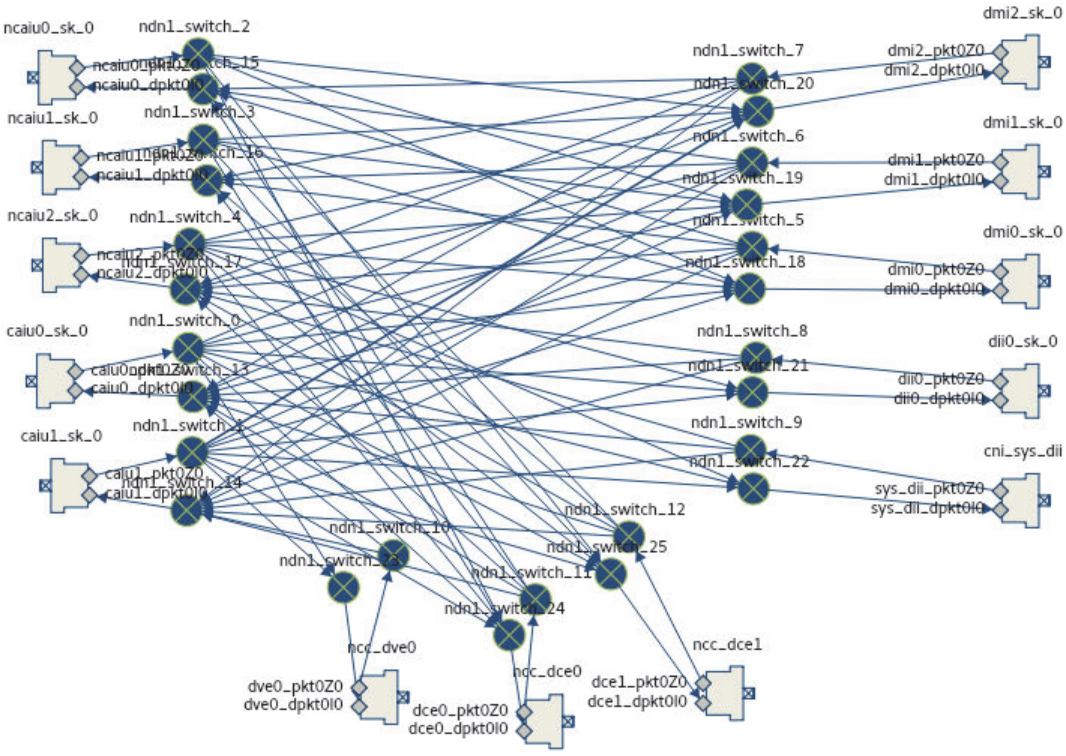
9.5.6 Create Topology

For this design, we will create one Manual and two Regular Topologies (see “Topologies”).

1. In Tasks View under *'Create a topology for Network "ndn1" by:'*, click Preparing it for Manual Editing.



2. Using the Topology editor, navigate to Topology Editor – 1. ndn1. It should look similar to the figure below. If the topology looks different, you can click and drag components around to arrange them similar to this figure.



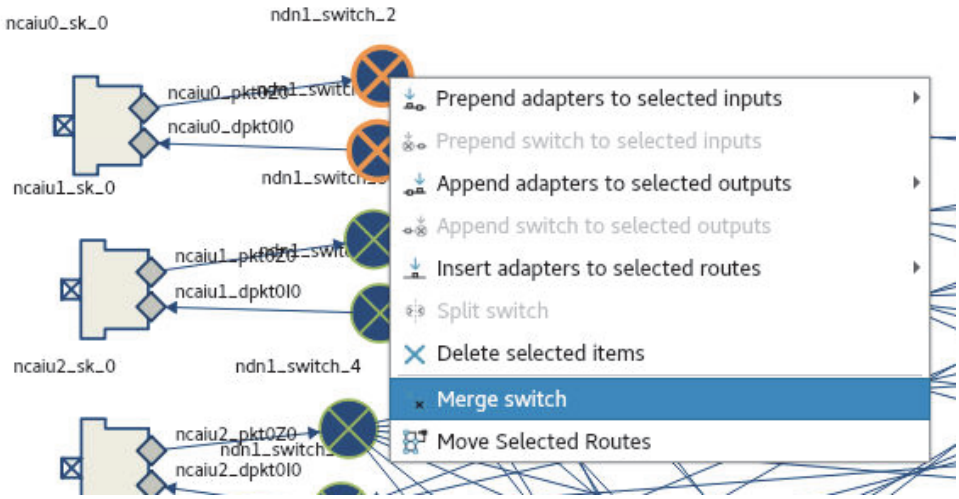
3. Start merging the input and output switches of each component together. There are two main ways of doing this: multi-select and the Merge switch toolbar button.

4. To use multi-select:

   a. Click on a switch — it should be highlighted in orange.

b. Holding Ctrl, click on another switch. Both switches should be highlighted.

c. Release Ctrl and right-click on one of the switches to see the following menu.



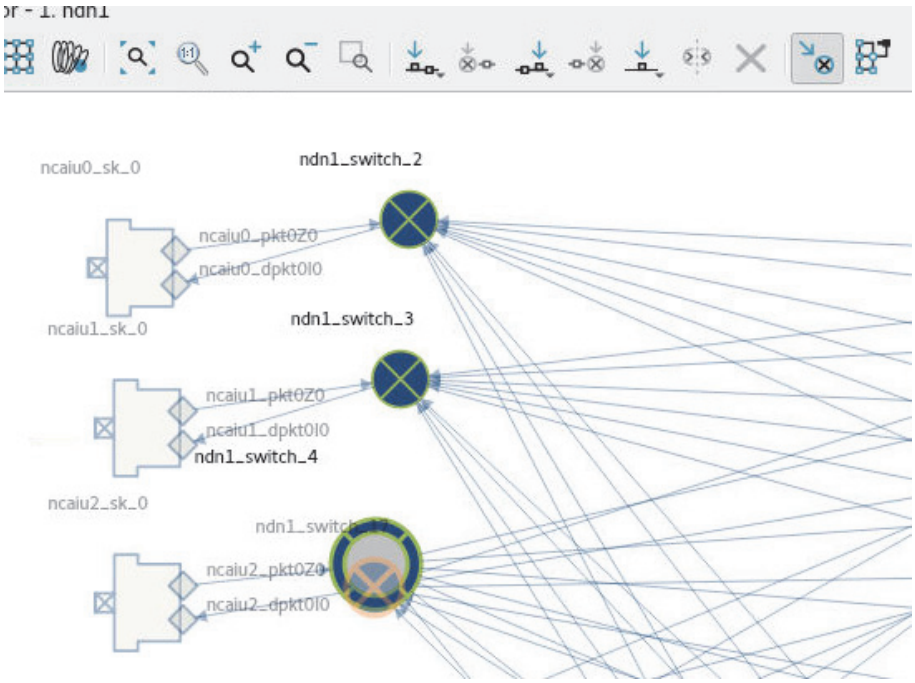
5. Select Merge switch.

Alternatively, after selecting two or more switches you can click the Merge switch button in the toolbar.

6. Merge switch toggle:

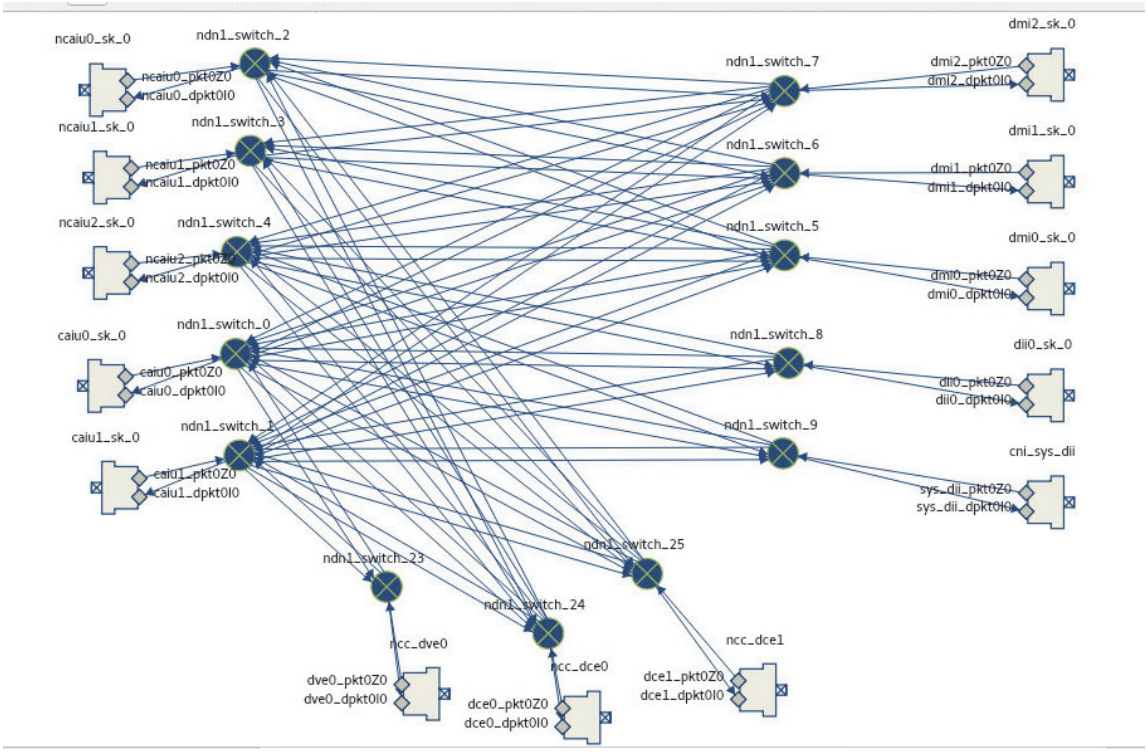
a. In the toolbar, select the Merge switch button (shown above). This will act as a toggle and put you in the Merge switch mode.

b. Click and drag a switch over top of a switch you would like to merge. It should look like the figure below.

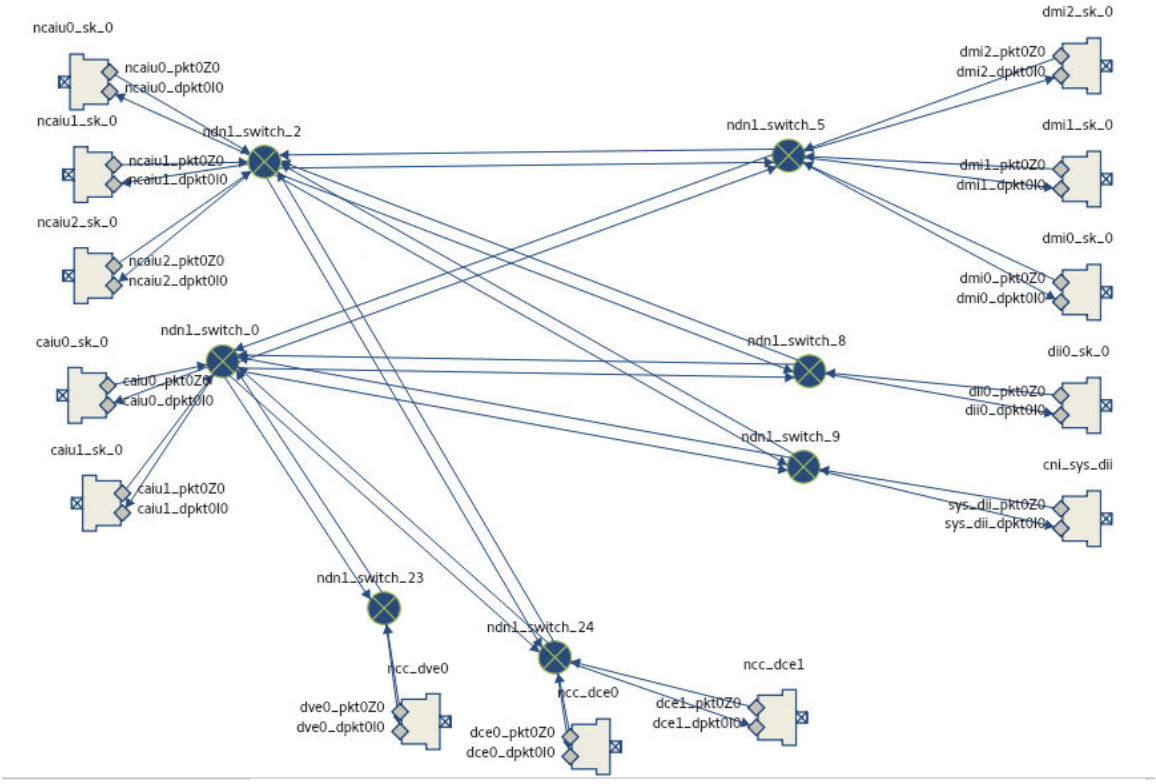


c. Release the switch. The two switches have now been merged.

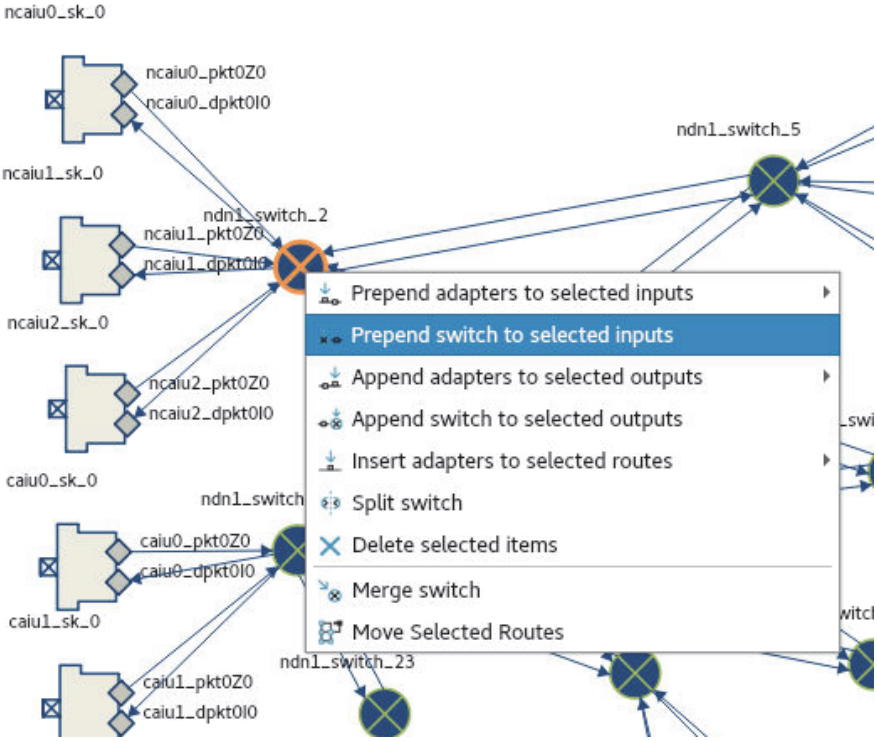
7. After merging the input and output switches together, you should have something similar to the figure below.



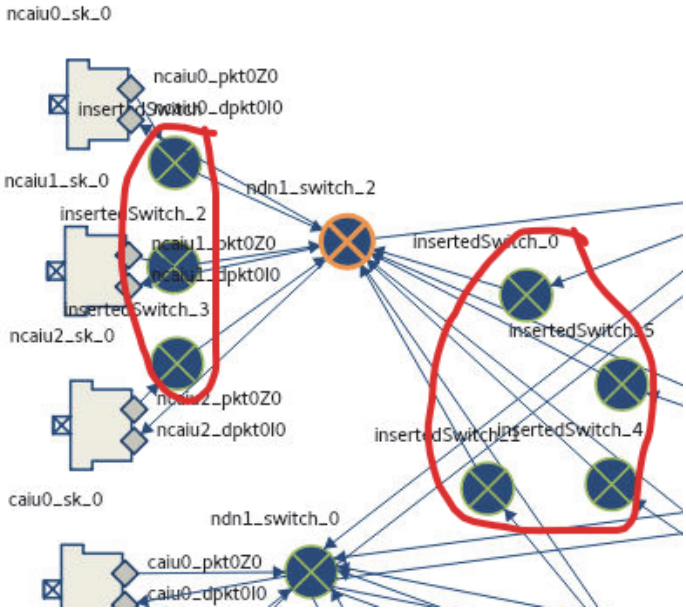
8. Merge the switches of the NCAIU into one switch. Repeat for CAIU, DCE, and DMI. Note that based on the order of merging, the names of the switches might be slightly different.



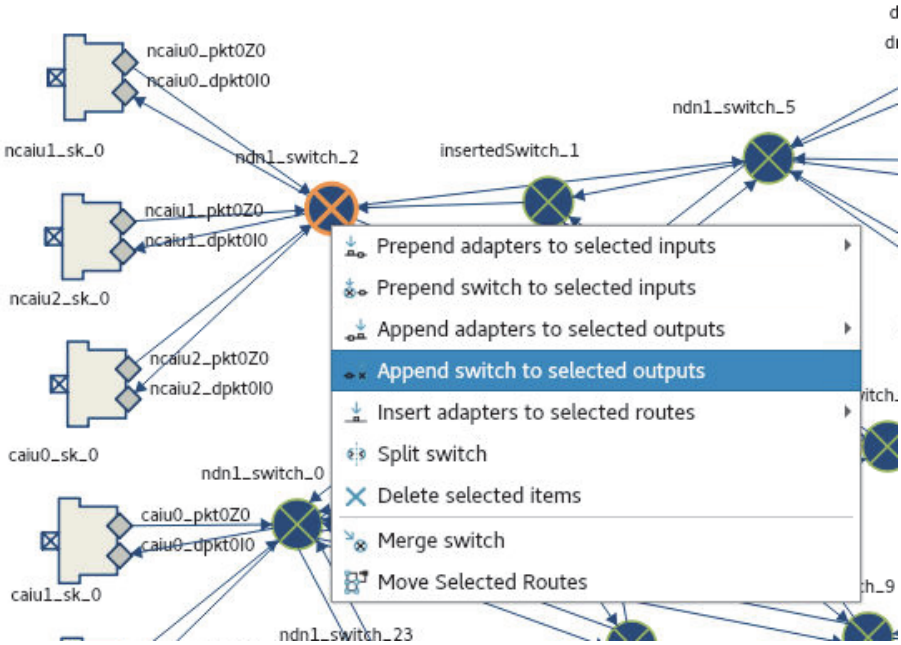
9. After selecting the switch in front of the NCAIUs, right click the switch and select Prepend switch to selected inputs.



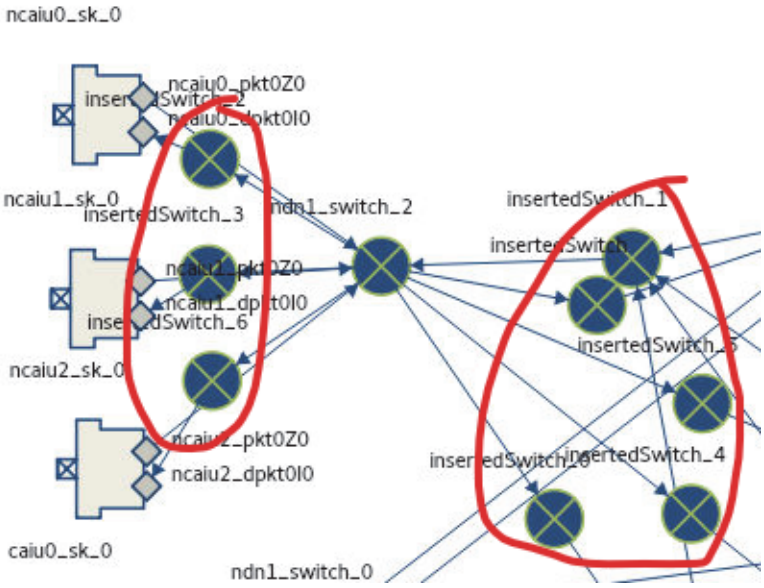
10. Merge the following switches together. After merging the 3 switches on the left together, merge the new switch back into the original switch, ndn1_switch_2 in this case.



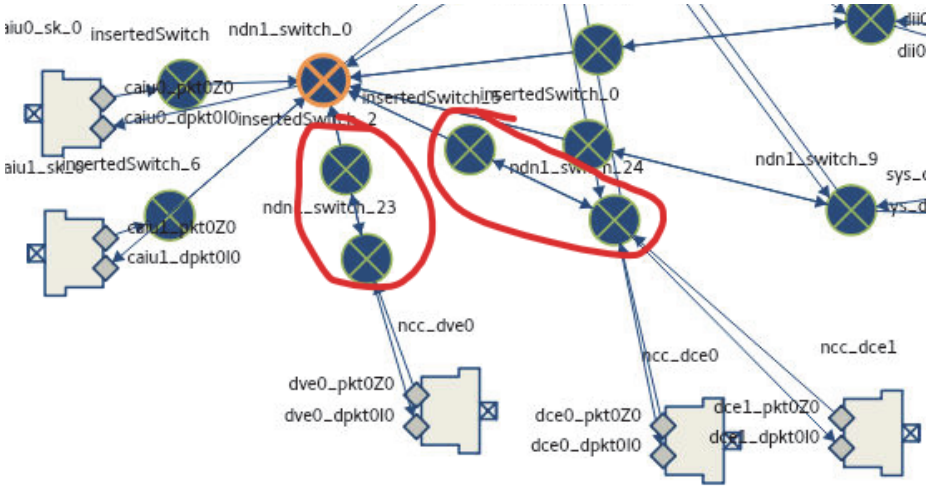
11. Select the switch in front of the NCAIUs and select the Append switch to selected outputs option.



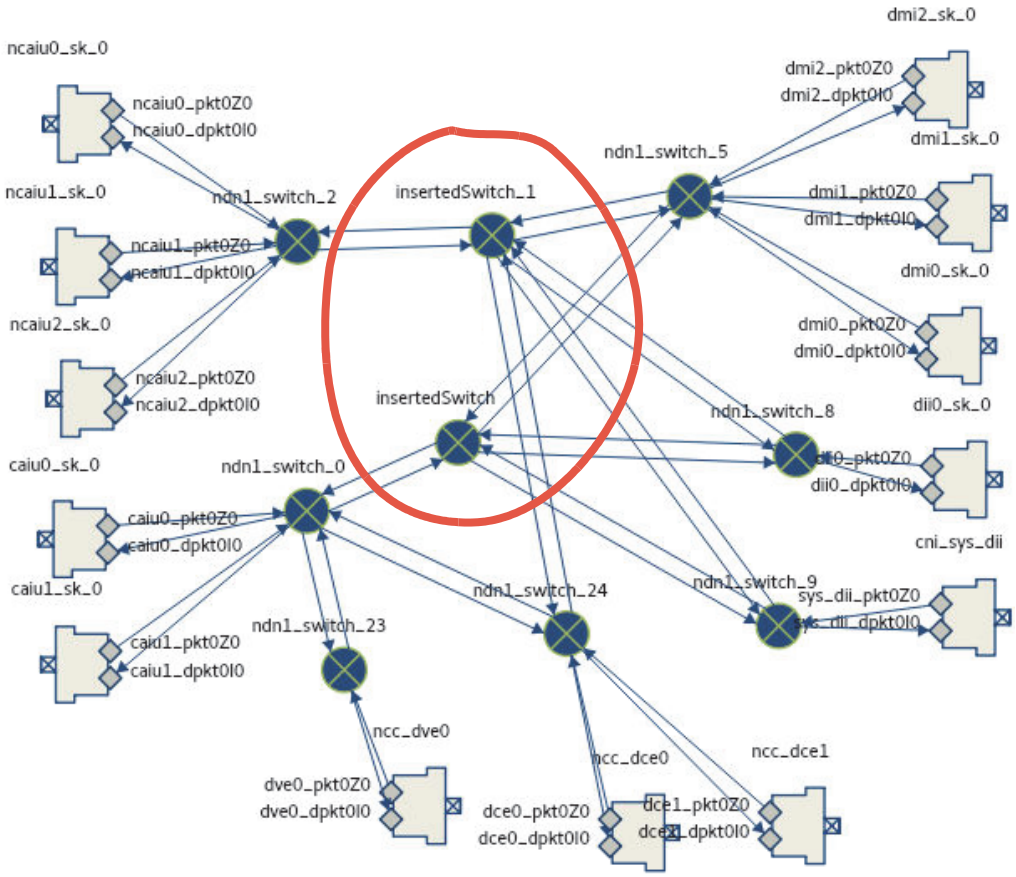
12. Merge the following switches together, merging the 3 switches on the left back into ndn1_switch_2 similar to the last step.



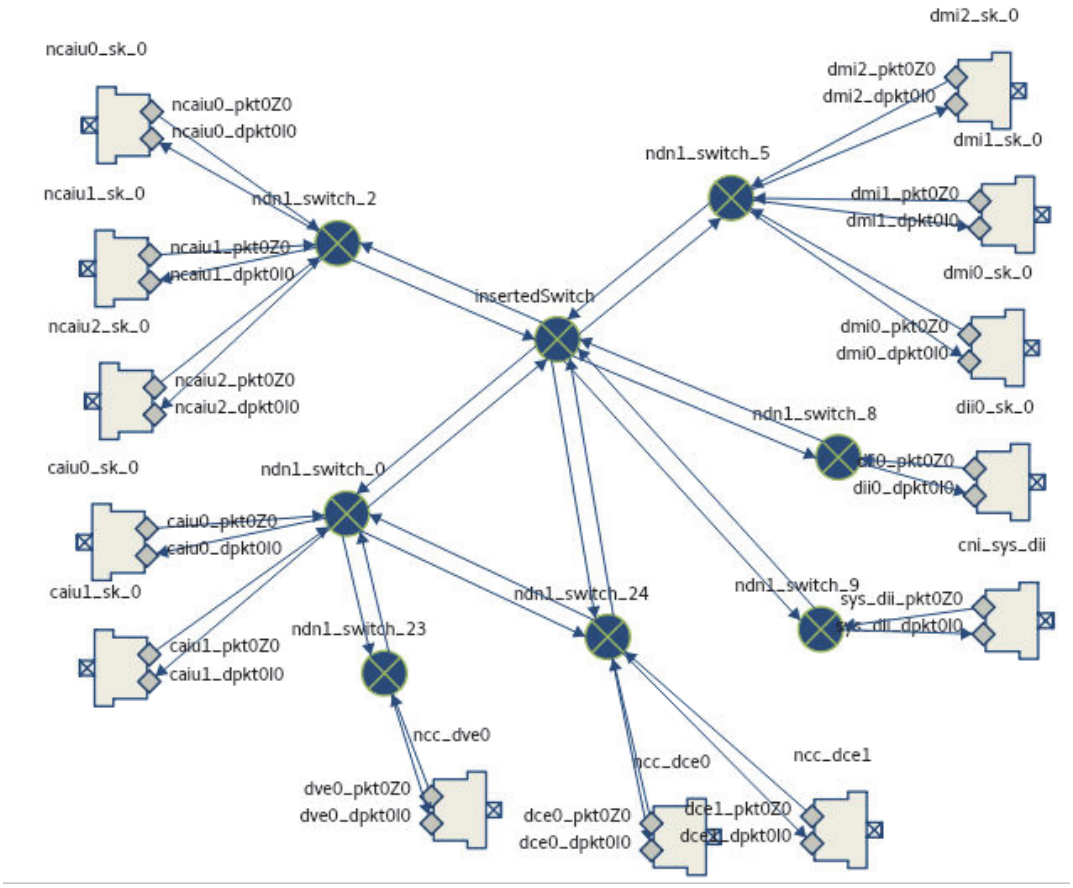
13. Repeat steps 9-12 on the switch in front of the CAIUs. For the switches that get added in front of the DVE and DCEs, merge them back into the DVE and the DCEs switch respectively as shown in the figure below.



14. After completing the previous step, you should have a Topology that looks like the figure below. Merge the two switches named InsertedSwitch together.

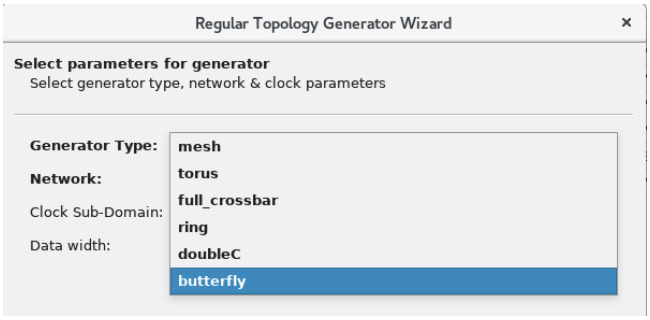


15. You have manually configured a topology.



16. Under ‘Create a topology for Network “ndn2” by:’, click “Making it a Regular Topology”.

17. For Generator Type, use the pulldown and select “butterfly”.



18. Accept the defaults, click Next, and click Finish.

19. Under ‘Create a topology for Network “dn” by:’, click “Making it a Regular Topology”.

20. For Generator Type, use the pulldown and select “butterfly” just as you did before.

21. Accept the defaults, click Next, and click Finish.

9.5.7 Insert Interrupt Handling

22. Go to Tasks View and click “Insert Interrupt Handling”.
23. Accept the defaults, and click OK.

    This runs an automation routine that creates and connects all interrupt pins. You can see the outcome of this operation by looking at the Architectural View, opening up the collapsed dock on the right of the editor (click the black “<” button), then clicking on the “Show Interrupt” button. This highlights in red the interrupt wires sent out of the toplevel. Note that two lines (maskable and non-maskable interrupts) come out of each Ncore Unit and pierce through the toplevel. Hovering on a pin will show the pin name in a tooltip.

9.5.8 Insert Configuration Network

24. Go to Tasks View and click “Insert Configuration Network”.
25. Accept the defaults, and click OK.

    This runs an automation routine that inserts the CSR network to connect all configuration registers. The Topology Editor will acquire two extra tabs for the request and response sides of the Configuration Network. the Project Tree will also show two extra Networks.

9.5.9 Insert Adapters

26. In Tasks View click “Insert Required Adapters”.
27. Accept the defaults, and click OK.

9.5.10 Set Parameters

28. Go to the Project Tree, navigate to system ->subsystem ->Architecture ->topology, and set the following parameters in Parameter View.

Parameter	Value
Name	topology
Number of DVM command request credits	**4**
Number of DVM snoop request credits	4
Memory Protection	NONE
Template	TwoCtrlOneDataTemplate
Resilience	☐
Native interface protection	☐
General purpose address regions	**6**
TimeOut threshold	16384
Enable QoS support	☐
QoS value to priority map	[]
QoS event threshold	16
System Directory primary set select bits	**[40]**
Enable HW assertions (Internal Parameter)	☐

29. For the Interface Units (under topology), set the following parameters before mapping. These are known as preMap parameters. See “Maestro Parameters” for more information. Important: Make sure to check or uncheck appropriate checkboxes for proxy cache enablement.

30. Set **dce0** parameters:

Number of DCE write request buffer cr...	2
Number of snoop credits	2
Number of memory read credits	2
Number of active coherent transactions	48

31. Set **dce1** parameters:

Number of DCE write request buffer cr...	2
Number of snoop credits	2
Number of memory read credits	2
Number of active coherent transactions	48

32. Set **caiu0** parameters:

Credits for Native Protocol	15
Outstanding transaction table Entries	48
Credits per DCE Command Request	8
Credits per DMI Command Request	4
Credits per DII Command Request	4
Stashing Snoop Credits	2

33. Set **caiu1** parameters:

Credits for Native Protocol	15
Outstanding transaction table Entries	48
Credits per DCE Command Request	8
Credits per DMI Command Request	4
Credits per DII Command Request	4
Stashing Snoop Credits	2

34. Set **ncaiu0** parameters:

Assigned SnoopFilter	project/chip0/sys0/subsys0/topology/sf1
Enable Proxy Cache	☑
Tag Banks	1
Data Banks	1
Cache Replacement Policy	RANDOM
Outstanding transaction table Entries	64
Credits per DCE Command Request	8
Credits per DMI Command Request	4

35. Set **ncaiu1** parameters:

Assigned SnoopFilter	project/chip0/sys0/subsys0/topology/sf1
Enable Proxy Cache	☑
Tag Banks	1
Data Banks	1
Cache Replacement Policy	RANDOM
Outstanding transaction table Entries	48
Credits per DCE Command Request	8
Credits per DMI Command Request	4

36. Set **ncaiu2** parameters:

Outstanding transaction table Entries	**32**
Credits per DCE Command Request	**4**
Credits per DMI Command Request	**4**
Credits per DII Command Request	**4**

37. Set **dmi0** parameters:

Number of DMI write buffers	**16**
Max outstanding memory read transactions (RTT Entries)	**48**
Max outstanding memory write transactions (WTT Entries)	**32**

38. Set **dmi1** parameters:

Number of DMI write buffers	**16**
Max outstanding memory read transactions (RTT Entries)	**48**
Max outstanding memory write transactions (WTT Entries)	**32**

39. Set **dmi2** parameters:

Number of DMI write buffers	**16**
Max outstanding memory read transactions (RTT Entries)	**32**
Max outstanding memory write transactions (WTT Entries)	**16**

40. Set **dii0** parameters parameters:

Number of Write request buffer credits	**16**
Max outstanding read transactions	**32**
Max outstanding write transactions	**16**

9.5.11 Phase 3 Checks

41. In Tasks View, click “Run Checks”. Make sure there are no errors (View Outstanding Issues).
42. If there are no issues, click “Move forward to Mapping/Mapper”.

9.6 Phase 4 - Mapping

Mapping is done automatically after you clicked “Move forward to Mapping/Mapper”. You should see messages scrolling in the window as the mapping process executes automatically.

1. In Tasks View, click “Run Checks”. Make sure there are no errors (View Outstanding Issues).

2. If there are no issues, proceed to Phase 6, Export.

9.7 Phase 5 - Refinement

This phase is not used for this design.

9.8 Phase 6 - Export

To complete the flow, communication with the server needs to occur to generate all collateral. The client plus server package is finalized in this phase.

Tasks for this phase:

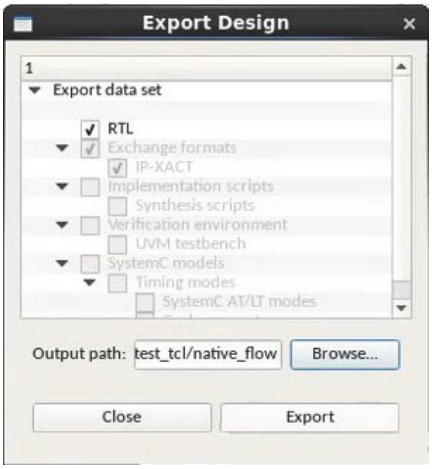
- "Generate RTL"

9.8.1 Generate RTL

1. In Tasks, click "Move forward to Export/Export".
2. Click Generate RTL.

   This can also be invoked with Flow -> Generate RTL or the toolbar icon.

3. Make selections for exporting the design.



4. When finished, click Export.

   In the Console, you should see the following message indicating a successful export.

```
Info: [REQ-40] Collateral generation is completed
Info: [AUT-902] Task export_design successfully performed
```

10 Maestro Parameters

This chapter describes the Ncore parameters used by Maestro.

TABLE 14. Maestro/Ncore Parameters

ID	Name	Type	Title	Description	Min	Max	Default
0	regionBaseAddr	Str	Boot Region Base Address	Specify boot region base address. This address must be aligned to the size specified.			0x0
1	regionHui	Int	Boot Region Home Unit ID	Specify the home unit ID associated with the boot region. If the home unit is a DII then specify the node ID of the DII. If the home unit is a DMI then specify the associated DMI interleaving group ID.			0
2	regionSize	Int	Boot Region Size	Specifies the size of boot region. Minimum is 4KB and must be a power of two.			4096
3	regionHut	Str	Boot Region Home Unit Type	Specify the home unit Type associated with the boot region. This can be either DMI (Memory) or DII (IO device).			DMI
4	Name	Str	Name of the signal	Name of the soc signal that is to be routed and connected to the memory instance.			
5	Direction	Str	Direction of the signal	Direction of the signal.			
6	Width	Int	Width of the signal	Width of the signal.	1	128	
19	power_ref	HierStr	Power Domain	Power Domain associated with this clock domain.			
20	gating	Str	Gating	The gating value determines whether the clock specified in the parent clock region is subject to gating when sent to a portion of the chip, that should be described as this clock domain. Select the option 'always_on' if no gating should be applied, or 'external' if logic external to the inter-connect can gate this clock.			always_on
21	power_ref	HierStr	Power Region	Power Region associated with this clock region.			
22	frequency	Int	Frequency	Frequency of the clock signal.	400000	2200000	500000
24	unitClockGating	Bool	Unit-Level Clock Gating	Whether it is desired that units receiving a clock signal from this Clock Subdomain should be subject to unit-level clock gating, when possible, for power savings. Disabling this optimization might help timing.			FALSE
25	frequency	Int	Frequency (kHz)	This parameter sets Clock frequency for the system.			300000
26	voltage	Int	Voltage (mV)	Voltage of the power supply of the Power region.			1000
54	nDves	Int	DVE Count	Total Number of DVEs in a given Ncore system.	1	1	

TABLE 14. Maestro/Ncore Parameters

ID	Name	Type	Title	Description	Min	Max	Default
60	gating	Str	Gating	Whether the voltage supply specified in the parent Power Region should be subject to disconnection when sent to a portion of the chip, that should be described as this Power Domain. If different gating logic can independently disconnect portions of the chip from the Power Region supply, each must be modeled as a separate Power Domain. Ungated supply should also be modeled as its own Power Domain.			always_on
61	voltage	Int	Voltage	Voltage of the power supply of the Power Region.	1	25000	950
73	enableDVM	Bool	Enable DVM	Add AC channel and enable DVM support.			FALSE
80	AxIdProcSelect-Bits	Int	Processors ID bits	specify processor ID bits.			
105	REQ_RSVDC	Int	Width of request RSVDC	Width of the request RSVDC/user bits.			0
107	wData	Int	CHI data bus width	Width of data on the CHI interface.			128
109	NodeID_Width	Int	Width of Node Id	Width of the Node ID of the CHI Interface.	7	11	7
110	wAddr	Int	CHI address bus width	Width of the address on CHI interface.	44	52	48
113	REQ_RSVDC	Int	Width of request RSVDC	Width of the request RSVDC/user bits.			0
115	wData	Int	CHI data bus width	Width of data on the CHI interface.			128
121	name	Str	Name of Interface	Name of the interface.			
124	direction	Str	Direction of Interface	Direction of the Evt Interface.			master
174	nTaggedMonitors	Int	Number of Tagged exclusive monitors	Specify the desired number of tagged exclusive monitor per DCE instance. Note that each DCE instance will always have a basic exclusive monitor.	0	8	0
185	useAtomic	Bool	Enable Atomic Engine	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.			FALSE
207	domain_ref	HierStr	Domain	Clock subdomain associated with this unit.			
214	nonCoherentMode	Bool	Non-Coherent Mode	Only applicable if the interface is AXI. Whether the traffic on the interface should be treated as non-coherent. If a Proxy Cache is used, the traffic is always treated as coherent.			FALSE
220	group_ref	HierStr	Group	Group			
227	wArId	Int	Width of Ar Id	Specify the width of AXI interface ArId bits.	1	20	6
228	wAwId	Int	Width of Aw Id	Specify the width of AXI interface AwId bits.	1	20	6
229	wAddr	Int	AXI address bus width	Specify the width of AXI interface address bits.	12	64	32
231	wData	Int	AXI data bus width	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.	8	2048	64.
233	generateHierarchy	Bool	Generate Hierarchy	Whether to generate a layer of logical hierarchy to the RTL.			

TABLE 14. Maestro/Ncore Parameters

ID	Name	Type	Title	Description	Min	Max	Default
241	dceInterleaving-PrimaryBits	Int	System Directory primary set select bits	Address bits to be used as interleaving bits for DCE selection.	6	64	6
245	upstream_ref	HierStr	Upstream Packet Port	Upstream packet port in a Link.			
246	downstream_ref	HierStr	Downstream Packet Port	Downstream packet port in a Link.			
262	primaryInterleav-ingBitOne	Int	Primary Interleaving Bit #1	Primary Address bit 1 to address DMI Interleaving. This bit forms the LSB in the vector of primary bits. Only this bit is set for a 2-way interleaving Function.	0	8192	0
264	primaryInterleav-ingBitTwo	Int	Primary Interleaving Bit #2	Primary Address bit 2 to address DMI Interleaving. In case of 4-way Interleaving Function, address bits should be set for primaryInterleav-ingBitOne and primaryInterleavingBitTwo.	0	8192	0
266	primaryInterleav-ingBitThree	Int	Primary Interleaving Bit #3	Primary Address bit 3 to address DMI Interleaving. In case of 8-way Interleaving Function, address bits should be set for primaryInterleav-ingBitOne, primaryInterleavingBitTwo and primaryInterleavingBitThree.	0	8192	0
268	primaryInterleav-ingBitFour	Int	Primary Interleaving Bit #4	Primary Address bit 4 to address DMI Interleaving. In case of 16-way Interleaving Function, address bits should be set for primaryInterleav-ingBitOne, primaryInterleavingBitTwo, primary-InterleavingBitThree and primaryInterleavingBitFour.	0	8192	0
279	memoryBase	Int	Base Address	The base address of the region.	0		
280	memorySize	Int	Size	Size of the memory. Each region size is power of 2, starting address is aligned to its size.	0		
339	domain_ref	HierStr	Domain	Clock subdomain associated with this unit.			
346	socket_ref	HierStr	Socket	Socket			
367	qosEnabled	Bool	Enable QoS support	Enable QoS capability.			FALSE
368	qosMap	Str	QoS value to priority map	4 bit Native interface QoS value map to 3 bit priority. value 0 has the highest priority and value 7 has the lowest priority.			
371	qosEventThresh-old	Int	QoS event threshold	QoS starvation threshold. Maximum number of high priority requests that can bypass a lower priority request.	1	8192	16
372	domain_ref	HierStr	Domain	Clock subdomain associated with this unit.			
383	domain_ref	HierStr	Domain	Clock subdomain associated with this socket.			
384	protocolType	Str	Protocol	Protocols supported by ARM.			<not set>
385	socketFunction	Str	Function	The function of the Socket is used to infer which type of Ncore Unit is deployed on the inside of the Subsystem.Allowable values are: - INITIATOR for any initiators, only for SLAVEs. - MEMORY for targets connected to memories, only for MASTERs. - PERIPHERAL for targets connected to peripherals, only for MASTERs.			INITIATOR
389	nGPRA	Int	General purpose address regions	Number of general purpose address regions that the system can support. Refer to the reference manual.	1	24	1

TABLE 14. Maestro/Ncore Parameters

ID	Name	Type	Title	Description	Min	Max	Default
391	resilienceEnabled	Bool	Resilience	Enable resilience-related features in the Ncore system. Refer to the reference manual for details on available features.			FALSE
392	duplicationEnabled	Bool	Unit duplication	Enable unit duplication for all Ncore units only. Memories and interconnect logic are not duplicated; they may be protected separately.			FALSE
393	nativeIntfProtEnabled	Bool	Native interface protection	Enable capability to add protection on native Ncore interfaces like CHI/ACE/AXI4. This adds an empty Verilog module with specified signals at the interface. Protection logic can be added in this Verilog module.			FALSE
395	coherentTemplate	Str	Template	Control and data network options: OneCtrlOneDataTemplate: Adds support for a single control and a single data network. TwoCtrlOneDataTemplate: Adds support for two control and a single data network. ThreeCtrlOneDataTemplate: Adds support for three control and a single data network. Refer to the reference manual for more details.			
396	interUnitDelay	Int	Duplicate unit delay	Delay between functional unit and delay unit. Delay can be specified in number of clock cycles.	1	4	1
398	resiliencyProtectionType	Str	Interconnect protection type	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.			NONE
399	memoryProtectionType	Str	Memory Protection	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.			NONE
406	timeOutThreshold	Int	TimeOut threshold	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.	1	2147483647	16384
407	nDvmCmdCredits	Int	Number of DVM command request credits	Maximum number of DVM command requests an AIU can have in flight.	1	4	1
408	nDvmSnpCredits	Int	Number of DVM snoop request credits	Maximum number of DVM snoop requests an AIU can have in flight on its native interface.			2
489	outputDomain_ref	HierStr	Output Clock Domain	The output Clock domain.			
495	domain_ref	HierStr	Clock Domain	Clock domain.			
496	depth	Int	Depth	The number of words in the SRAM. Can be 4 or greater.	4	1048576	16
501	latency	Int	Latency	Number of clock signals needed to assert error signals out of the UDV module.	0	8192	1
507	wireName	Str	Port Name	Port Name for Generic port.			
508	wireWidth	Int	Port Width	Port width for generic port.	0	1024	
510	wireSignaling	Str	Port Signaling	Port Signaling.			LEVEL
511	wireUpf	Str	Port UPF Value	Port UPF value.			

TABLE 14. Maestro/Ncore Parameters

ID	Name	Type	Title	Description	Min	Max	Default
514	domain_ref	HierStr	Domain	The clock domain associated with the Interrupt Accumulator.			
519	snoopFilter_ref	HierStr	Assigned SnoopFilter	Specify the snoop filter associated with this AIU.			
520	nNativeCredits	Int	Credits for Native Protocol	Specify the maximum number of CHI link credits this AIU should support	2	15	2
521	nOttCtrlEntries	Int	Outstanding transaction table Entries	Specify the maximum number of outstanding native transactions this AIU should support.	4	128	4
522	nDceCmdCredits	Int	Credits per DCE Command Request	Specify the maximum number of credits for coherent transactions per DCE. This should be determined based on required bandwidth and network round trip latency.	2	16	2
523	nDmiCmdCredits	Int	Credits per DMI Command Request	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.	2	16	2
524	nDiiCmdCredits	Int	Credits per DII Command Request	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.	2	16	2
525	nStshSnpCredits	Int	Stashing Snoop Credits	Specify the maximum number of outstanding stash snoops this AIU should support. These are stash snoops issued on the CHI interface.	1	8	2
527	nDceRbCredits	Int	Number of DCE write request buffer credits	Specify the maximum number of DCE write request buffer credits per DMI. These credits limit number of Coherent writes and includes snoops that can cause a write to DMI.	2	16	2
528	nAiuSnpCredits	Int	Number of snoop credits	Specify the maximum number of snoop request credits per AIU.	2	16	2
529	nDmiMrdCredits	Int	Number of memory read credits	Specify the maximum number of memory read credits per DMI.	2	16	2
530	nAttCtrlEntries	Int	Number of active coherent transactions	Specify the maximum number of active coherent transactions tracked by each DCE.	4	64	4
531	nDiiRbCredits	Int	Number of Write request buffer credits	Specify the maximum number of non coherent write request buffer credits.	2	32	2
532	nRttCtrlEntries	Int	Max outstanding read transactions	Specify number of outstanding read transactions on the AXI interface.	4	32	4
533	nWttCtrlEntries	Int	Max outstanding write transactions	Specify number of outstanding write transactions on the AXI interface.	4	32	4
534	nDmiRbCredits	Int	Number of DMI write buffers	Specify the maximum number of non coherent write request buffer credits.	2	64	2
535	nRttCtrlEntries	Int	Max outstanding memory read transactions (RTT Entries)	Specify number of outstanding read transactions on the AXI interface.	4	128	4
536	nWttCtrlEntries	Int	Max outstanding memory write transactions (WTT Entries)	Specify number of outstanding write transactions on the AXI interface.	4	64	4
538	hasProxyCache	Bool	Enable Proxy Cache	This options enables Proxy Cache support. This is supported only with AXI interface.			FALSE

TABLE 14. Maestro/Ncore Parameters

ID	Name	Type	Title	Description	Min	Max	Default
539	nOttCtrlEntries	Int	Outstanding transaction table Entries	Specify the maximum number of outstanding native transactions this AIU should support.	4	128	4
540	nDceCmdCredits	Int	Credits per DCE Command Request	Specify the maximum number of credits for coherent transactions per DCE. This should be determined based on required bandwidth and network round trip latency.	2	16	2
541	nDmiCmdCredits	Int	Credits per DMI Command Request	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.	2	16	2
542	nDiiCmdCredits	Int	Credits per DII Command Request	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.	2	16	2
565	depth	Int	Depth	Depth of the Pipe Adapter.	1	2	2
573	outputDomain_ref	HierStr	Output Clock Domain	Clock domain associated at the output port of the rate adapter.			
574	width	Int	Data Width	Data width of the SMI Port.	0	8192	32
575	buffers	Int	Number of Buffers	Number of buffers in the SMI Port.	0	8192	0
576	messagetypes_ref	HierStr	Message Types	Reference to the message types.			
612	targ_id	Str	targ_id	Target Id.			
613	route	Str	Route	Route.			
614	name	Str	Instance name	Instance name.			NONE
619	syncDepth	Int	Synchronizer depth	Synchronizer depth - i.e., pipe stages for synchronizer.	2	10	2
636	wireRtlPrefix	Str	RTL Prefix	RTL Prefix. For a given block, all the ports must have the same writeRtlPrefix.			
637	channel	Str	channel	Channel for PMA.			"Q"
649	targ_id	Str	targ_id	Target Id.			
650	egress	Int	Egress Port	Egress port.			
713	wLateClk	Int	Width of Late Clock	Width of the late clock in the fault interface.	0	1	0
715	SrcID	Int	Width of source Id	Width of the Source ID.	7	11	7
716	TgtID	Int	Width of target Id	Width of the Target ID.	7	11	7
717	enPoison	Bool	Enable poison bits	Enable poison port support on CHI interface.			FALSE
718	wAwUser	Int	Width of Aw user	Specify the width of AXI interface Aw user bits.	0	32	0
720	wArUser	Int	Width of Ar user	Specify the width of AXI interface Ar user bits.	0	32	0
721	wAddr	Int	width of address	Width of the address bits.			32
730	SrcID	Int	Width of source Id	Width of the Source ID.	7	7	7
731	TgtID	Int	Width of target Id	Width of the Target ID.	7	7	7
732	wAddr	Int	CHI address bus width	Width of the CHI address bus.	44	44	44
733	wAddr	Int	width of address	Specify the width of AXI interface address bits.	12	64	32
734	wArUser	Int	width of ar user	Specify the width of AXI interface Ar user bits.	0	200	0
735	wArId	Int	Width of ar id	Specify the width of AXI interface ArId bits.	0	32	0

TABLE 14. Maestro/Ncore Parameters

ID	Name	Type	Title	Description	Min	Max	Default
736	wAwUser	Int	Width of aw user	Specify the width of AXI interface Aw user bits.	0	200	0
737	wAwId	Int	Width of aw id	Specify the width of AXI interface AwId bits.	0	32	0
740	wData	Int	Width of data	Specify the width of AXI interface data bits.	8	2048	32
747	wQos	Int	Width of QoS	QoS port gets enabled as standard 4 bit wide port when QoS is enabled.	0	4	4
751	nSets	Int	Number of sets for the selected snoop filter	Number of sets for the selected snoop filter. This value should be a power-of-2.	16	1048576	16
753	nVictimEntries	Int	Number of victim buffer entries per DCE for the specified snoop filter.	The number of victim buffer entries for the tag filter.	0	64	2
754	aPrimaryBits	Int	Set Select Bits	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].			
765	inputBufferDepth	Int	Input buffer depth	Input buffer depth of switch.			0
768	subsystemType	Str	Subsystem Type	Subsystem is a self-standing portion of the architectural design. The enumerated values define which architecture is selected for the design. 'ARTERIS_COHERENT' design supports cache-coherency. 'ARTERIS_NONCOHERENT' design does not support cache coherency. 'ARTERIS_LLC' supports Last Level Cache.			<Undefined>
771	nMainTraceBufSize	Int	Number of trace entries in the buffer	Specifies the number of trace entries in the buffer.	32	1024	64
772	nTraceRegisters	Int	Number of Trace Registers	Specifies the number of trace register sets present in the system per AIU.	1	4	1
791	productVersion	Str	Product Version	Product (Maestro) version.			0.0.0
792	projectLicenseName	Str	Project License Name	Project License Name.			
793	memoryType	Str	Memory Type	This parameter is used to set the type of memory. This parameter is an enumerated type whose values are either FLOP or SRAM.			FLOP
806	nPerfCounters	Int	Number of performance counters	Number of performance counters per Ncore unit.	4	8	4
808	packetSize	Int	Buffer Size	Size of multicast packet, in beats, to be buffered in block. Does not have to be entire packet.	1	1024	2
844	nPerfCounters	Int	Number of performance counters	Number of performance counters per Ncore unit.	4	8	4
850	nPerfCounters	Int	Number of performance counters	Number of performance counters per Ncore unit.	4	8	4
856	incrementalOpt	Bool	Incremental Optimization	Enables DC to perform incremental optimization. This works only on mapped logic.			TRUE
857	ulvtPercentage	Int	Ulvt Percentage	% of ulvt cells can be used by the tool.	0		0

TABLE 14. Maestro/Ncore Parameters

ID	Name	Type	Title	Description	Min	Max	Default
860	compileCommand	Str	Compile command	DC mapping and optimization command. Default: compile_ultra -spg -no_boundary_optimization -gate_clock			(see Desc.)
861	nProcessors	Int	Number of Processors	Specify the number of logical processors connected to this interface.			1
862	nPerfCounters	Int	Number of performance counters	Number of performance counters per Ncore unit.	4	8	4
866	fnCsrAccess	Bool	Enable CSR Access	Enables CSR access via this AIU.			TRUE
868	nPerfCounters	Int	Number of performance counters	Number of performance counters per Ncore unit.	4	8	4
869	fnCsrAccess	Bool	Enable CSR Access	Enables CSR access via this AIU.			TRUE
871	multicastLabels	Int	Carried Multicast Labels	Carried Multicast labels	0	8192	0
872	packetSize	Int	Buffer Size	Size of multicast packet, in beats, to be buffered in block. Does not have to be entire packet.	1	1024	2
878	nPerfCounters	Int	Number of performance counters	Number of performance counters per Ncore unit.	4	8	4
996	headerDepth	Int	Header Depth	The number of headers that the FIFO can hold. If headerSplit = false, then the number of headers does not apply.	1	1024	1
1046	rtlPrefixString	Str	Memory RTL name	Specify the name of the embedded memory module to instantiate. The system designer must provide the module to be instantiated with this name.			
1047	MemType	Str	Memory Type	Specify the choice of memory implementation, between a Register (flip-fl0p) implementation, or an embedded memory IP core instantiation. Ncore uses Synopsys embedded memory port names and timing.			NONE
1056	nSets	Int	Number of sets	Specify the number of sets/entries in the Cache.			16
1057	nWays	Int	Number of ways	Specify the number of ways/associativity of the cache.	2	16	2
1058	useScratchpad	Bool	Enable Scratchpad	This option enables scratch pad capability which is available only in DMI.			FALSE
1064	PriSubDiagAddrBits	Int	Primary Set Select Bits	Specify address bits to be used as primary set select bits.			
1065	TagBankSelBits	Int	Tag Bank Select Bits	Specify tag bank select bit. This bit must be one of the bits from the primary select bits.			
1066	DataBankSelBits	Int	Data Bank Select Bits	Specify data bank select bit. These bits must be from the primary select bits.			
1092	nLargestEndPoint	Int	Size of the largest end point	Specify the size of the largest endpoint device connected to the DII. The size is in KBs. This size is used to achieve endpoint ordering as defined by ARM CHI specification.	4	549755813888	4
1095	nAddrTransRegisters	Int	Number of Address Translation Registers	Specifies the number of address translation registers that are available within the DII. Refer to the address translation capability section in the reference manual.	0	4	0

TABLE 14. Maestro/Ncore Parameters

ID	Name	Type	Title	Description	Min	Max	Default
1103	nAddrTransRegisters	Int	Number of Address Translation Registers	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.	0	4	0
1105	nWayPartitioningRegisters	Int	nWayPartitioningRegisters	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.	0	16	0
1297	channel_ref	HierStr	Target Socket	Region maps to one target (or an aggregate of multiple identical targets for address striping).			
1298	mg_ref	HierStr	Target DynamicMemoryGroup	Defines the memoryGroup this bootregion can be accessed from.			
1496	headerSplit	Bool	Split Header Buffer	Split header buffer.			FALSE
1534	hasSysMemCache	Bool	SMC Enable	This option adds a System Memory Cache in DMI. It must be enabled when an atomic capable master is present in the system and requires at least a 4KB SMC.			FALSE
1535	nTagBanks	Int	Tag Banks	Number of tag banks in the System Memory Cache.			1
1536	nDataBanks	Int	Data Banks	Number of data banks in the System Memory Cache.			1
1540	cacheReplPolicy	Str	Cache Replacement Policy	Cache Replacement Policy.			RANDOM
1735	nTagBanks	Int	Tag Banks	Number of tag banks in the Proxy Cache.			1
1736	nDataBanks	Int	Data Banks	Number of data banks in the Proxy Cache.			1
2043	checkOnly	Bool	Compile and Link Only	Whether to stop the RTL flow after compilation and linking, or to proceed to synthesis.			
2044	topoMode	Bool	Topographical Mode	Whether to launch the synthesis tools in topographical mode, or WLM. The latter is faster but less accurate.			
2045	techNode	Str	Technology	The technology node for the design.			CUSTOM
2046	clockUncertainty	Int	Clock Uncertainty	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.	1	99	15
2047	rtlWrapperDir	Str	RTL Wrapper Dir	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.			
2048	hardMacroDbs	Str	Hard Macros	List of DB files containing compiled models of hard macros instantiated from RTL wrappers.			
2049	bottomUpSynthesis	Bool	Bottom-Up	Characterize and compile major functional blocks (subdesigns) first, and then the toplevel.			TRUE
2050	maxTransition	Int	Max Transition	Max signal transition time constraint, in ps.	0		150
2051	outputLoad	Int	Output Load	Capacitive load for all primary outputs, in fF.	0		100000

TABLE 14. Maestro/Ncore Parameters

ID	Name	Type	Title	Description	Min	Max	Default
2543	region_ref	HierStr	Physical Region	The physical region associated with this group.			
7551	portDataWidth	Int	Port Data Width	The data width of the packet port in a switch.	0	8192	0
7554	portPacketSerial-ization	Str	Port Packet Serialization	This parameter sets the packet style on the packet port.			HEAD-ER_PAR-ALLEL
10710	csrAccessSup-ported	Bool	CSR Access	Whether this Socket will be allowed access to the CSR network, if one is inserted. Only applicable to INITIATOR Sockets.			TRUE
25443	preferredCSRdo-main_ref	HierStr	Preferred CSR clock	Preferred CSR clock domain.			

11 Glossary

Table 15. Terms and Abbreviations

Terms	Descriptions
ATU	Arteris Transport Unit used in Non-Coherent Transport Interconnect.
C-AIU	Coherent Agent Interface Unit. Can be CHI-A, CHI-B, or ACE. C-AIU provides native interface for connecting CHI-A/CHI-B processors (processors compatible with ARM interface standards like Coherent Hub Interface Issue-A and Issue-B) with the Ncore 3 Coherency system. Used interchangeably with CHI-AIU.
Chip	Object to store the design that the user is trying to build.
Clock Adapter	Unit to synchronize clock frequencies.
Clock Domain	One of the main clock signals.
Clock Region	Collection to hold a group of clock signals.
Clock Subdomain	Clock derived from one of the main clocks.
CNI	Coherent Network Interface, used internally in Maestro.
ConfigSocket	Socket used for the Configuration Networks.
Database	Contains all the information entered by the user for a project, as well as information regarding the state of the tool.
DCE	Model for Distributed Coherency Engine. It is the central block in the coherence system and is responsible for maintaining memory coherence across the set memory locations mapped to it.
Depacketizer	Converts the packets back to native layer transactions. Packetizer is present in the Coherent Network Interface for every Ncore unit (see Packetizer).
DII	Model for Distributed IO Interface. The DII unit is a configurable IP block which acts as a gateway to the I/O devices.
DMI	Model for Distributed Memory Interface Unit. It is a configurable IP block which interfaces with the next level memory hierarchy (e.g., system memory).
DVE	The Distributed Virtual Memory Management Engine is the central block in the Ncore system for Distributed Virtual Memory transactions and is the point of serialization for all the DVM requests sent to it.
DVM	Distributed Virtual Memory.
Dynamic Memory Group	Container for DMI sockets. It can hold maximum of 4 DMI sockets.
Fabric	A set of interconnected packets, switching, and transport hardware units.
Generator	Engine in Maestro used to carry out a functionality.
GRB	The Global Register Block manages memory address.

Table 15. Terms and Abbreviations

Terms	Descriptions
GUI	Graphical User Interface. It is the software module which provides visual representation of items and enables the user to interact with them.
IO-AIU	Model for Non Coherent Agent Interface Unit. The IO-AIU serves the protocol agent using its native AXI/ACE interface and exchanges protocol messages with other blocks in the system using the internal messaging protocol.
IP	Intellectual Property refers to the hardware components delivered as RTL to customers who use them to build ASIC. ArterisIP products are considered as IP.
IP Generator (IPG)	The IPG concept is used by the Maestro user to access high-level functions. An IPG instance contains information usually filled by the user and/or through automation.
IP Generator Unit	The IP Generator unit is a basic element reflecting the structure of the IP created by the IP Generator.
Memory Map	Group of memory sets in the design. Only one active memory map is possible in the design.
Memory Set	Set of Memory Groups. The memory groups in the set must include all the DMIs.
MPF	Maestro Project files.
NC-AIU	Non-Coherent Agent Interface Unit
NCNI	Non-coherent Network Interface.
Network	The Network is a combination of switches connected through links, and other internal elements such as buffers and pipelines. Its function is to transport requests and responses between sources and destinations at the edge.
NoC	Network on Chip can be described as a class of communication fabrics. Arteris IP interconnects are based on NoC designs.
Object	Describes a user-visible container of information elements, organized in a structured manner, as well as a set of methods to interact with the information.
Packet port	A port in the design, used to send packets.
Packetizer	Converts the native layer transactions into packets to be transported into the network. It is present in the Coherent Network Interface for every Ncore unit (see Depacketizer).
Port	A simple input or output of a hardware unit. They can have multiple bits.
Power Domain	Power source for the Ncore unit.
Power Region	Collection of Power Domains.
Project	Container of all information required to create a Communication Subsystem. All elements of a projects are related to each other. Maestro lets the user work on one project at a time.
Route	Route is an ordered list of transport units where traffic can go through from a source Network Interface Unit (NIU) to a destination NIU. Routes uniquely define the set of physical and virtual connections in a network.
Routing	Contains information describing the connectivity between sockets, the address decode, and the transformation function used to route requests between them.
Session	The time duration when the Maestro tool is running continuously on a computer. During a session, information entered in the tool is assumed to be kept in memory.
SMI Port	Symphony Message Interface Port.

Table 15. Terms and Abbreviations

Terms	Descriptions
SoC	System on a Chip can be described as a class of Integrated Circuits built by assembling multiple components (CPU, memory, accelerators, etc.) around a communication subsystem.
Socket	A bundle of inputs and/or outputs of a hardware unit, that follows a communication protocol. Example of Sockets protocols are AMBA/AXI, OCP-IP, TileLink.
Subsystem	Any combination of ArterisIP products including a single instance of a particular IP and last level cache. The user creates and optimizes the subsystem using Maestro. Subsequent to RTL generation, the subsystem is used in the SoC.
Switch	Entity in a network which receives packets and forwards it to the destination unit.

Index

A

B

C

D

E

F

G

H

I

L

M

N

O

P

R

S

T

U

V

W

www.arteris.com
General Information:
U.S.A: (408) 470-7300
France: +33 1 61 37 38 40
Korea: +82 (70) 4849-2867
China: +86 139 1693 9647
Japan: +81 80 4683 5530
Headquarters – Campbell
595 Millich Drive, Suite 200
Campbell, CA 95008

ARTERIS IP