ARTERIS IP

Ncore 3.4 User Guide

ARTERIS.COM

Doc Name	Revision	Date	Description
100023_UG_3.4_1.0_A.1	3.4_1.0_A.1	Apr 13, 2023	Initial Release

Contents

1 Introduction

Arteris Ncore® is a configurable, scalable interconnect IP product that enables the design of high-performance heterogeneous SoCs by integrating fully coherent, IO coherent, and non-coherent agents into complex SoCs.

Ncore supports AMBA CHI and AMBA ACE fully coherent agents simultaneously, AMBA ACE-Lite IO-coherent agents, and AMBA AXI4 non-coherent agents. The non-coherent architecture may be further complemented using the Arteris FexNoC non-coherent IP product.

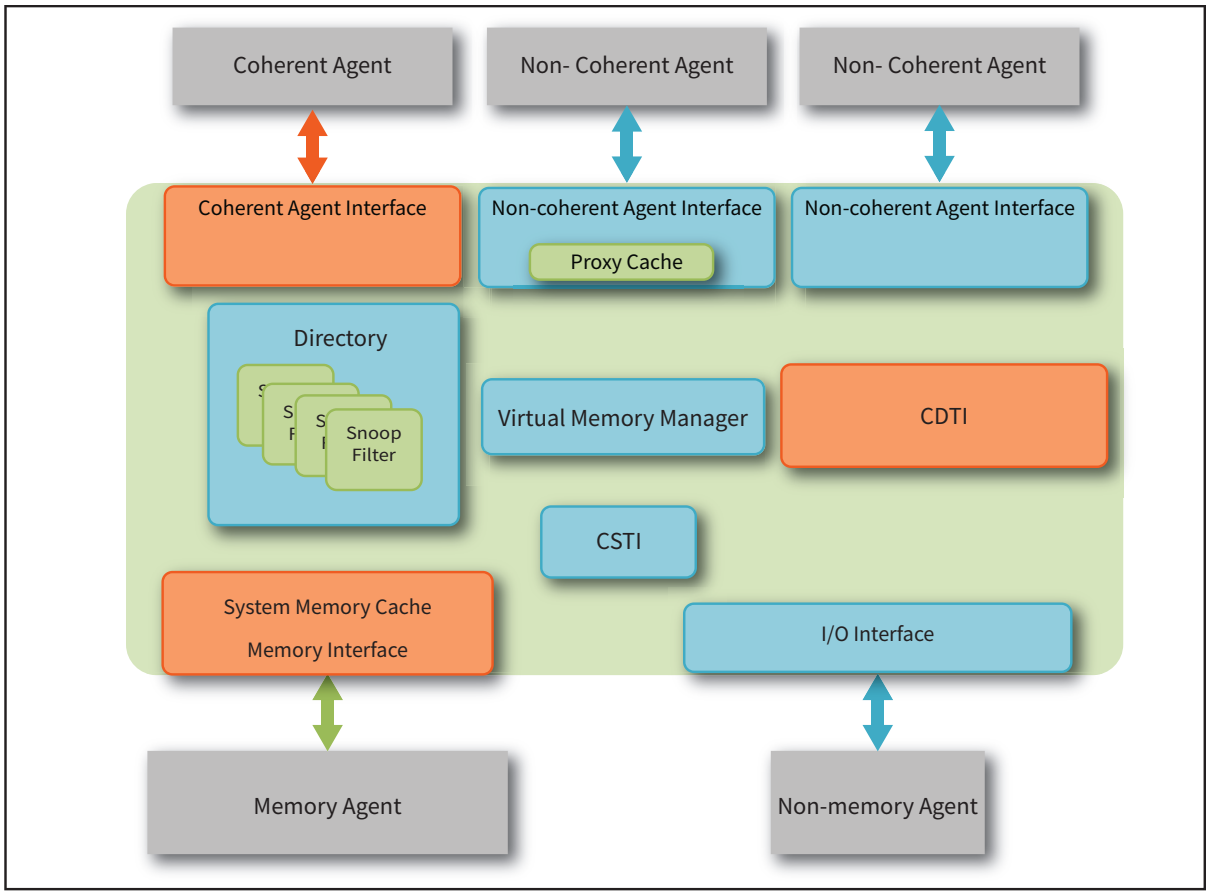
Additionally, the Ncore Resilience Package provides functional safety features for ISO 26262 qualification. See the *Ncore Reference Manual* for a detailed description of the interconnect architecture and hardware features.

Ncore is configured, built and optimized using the Arteris Maestro GUI, which guides users through specification, design, validation and output of SoC design collateral.

This user guide outlines the Maestro SoC workflow phases, from architecture to export, and includes a Demonstration Design that users may find useful to complete as an introduction to the flow.

Figure 1 shows the elements of an Ncore-based SoC.

Figure 1. Ncore Functional View



1.1 Features

Ncore supports an extensive feature set and configurability options to optimize SoCs across a wide gamut of performance, power, bandwidth and application requirements.

The Ncore features include:

- Up to 64 (CAIU) coherent ports supporting either AMBA CHI-A, AMBA CHI-B or AMBA ACE protocols
- Configurable distributed Snoop Filters (SF)
- Up to 64 (NCAIU) configurable as IO coherent ACE-Lite ports, or non-coherent AXI4 ports (AXI4 with Proxy cache option)
- Up to 16 (DMI) memory ports, supporting AXI4, address interleaving and System Memory Cache / Scratchpad options
- Up to 16 (DII) peripheral ports supporting AXI4
- Up to 16 (DCE) directory ports for coherency control
- Configuration Space Registers (CSR) accessed as an Ncore3 DII, and optionally an AMBA APB interface externally
- Configurable memory map options, supporting DMI interleaving and run-time options
- Boot region configuration and remap
- DCEs, DMIs and groups of initiators can interleaved with each other, dramatically reducing the network connectivity
- Support for ASIL Functional Safety standards
- Error reporting
- Performance monitors for latency, bandwidth and debug
- Multiple clock domains
- Highly configurable proprietary coherent interconnect topology
- Configurable number of networks (3 or 4), and data network widths (64, 128 and 256bit data)
- Topology can be selected and optimized per network, interactively in Maestro
- High performance, scalable solution 1.6 GHz in a 16 FFC process

Maestro exports include:

- Verilog RTL
- Synthesis scripts
- Extensible UVM test environment
- System C models

- IP-XACT
- Synthesis flow constraints
- CSR space register documentation

1.2 About the Project File

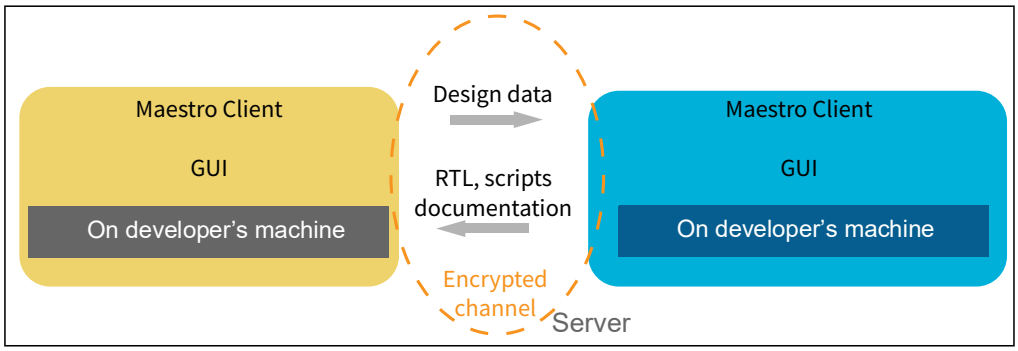
Ncore stores each design in a Maestro Project File (MPF). The file contains all of the design data required by Ncore for a specific project.

1.3 Design Configuration Workflow

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

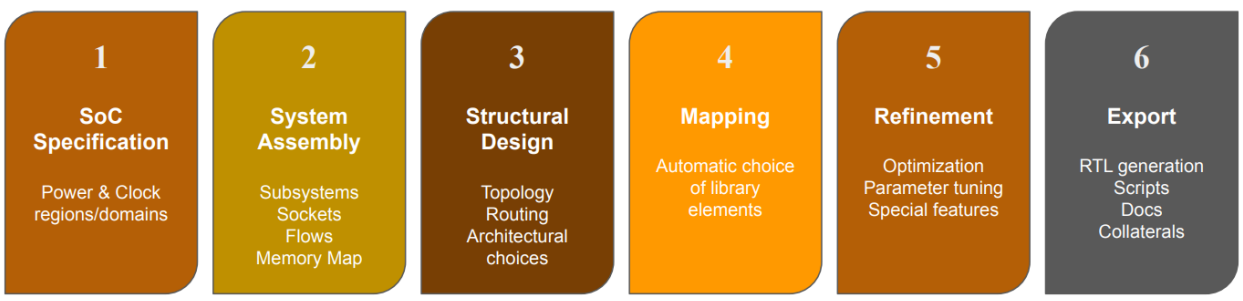
- The client includes all designer-visible functionality from specification through to refinement and design checking.
- The server component, which can be run at Arteris or on-premises depending on the license agreements, is invoked during the design workflow to generate RTL and collaterals as listed above.

Figure 2. Maestro client-server component



1.3.1 Ncore Workflow Phases

The Ncore workflow encompasses a sequence of design actions known as phases as shown in the following figure.



You perform the design actions and are assisted by Maestro.

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. See "SoC Specification (Phase 1)". Setting the safety ASIL value is also part of this phase.
2. Phase 2 - The System Assembly phase of the design flow defines sockets, connectivity requirements and memory maps. See "System Assembly (Phase 2)".
3. Phase 3 - The Structural (Architectural) Design phase of the design flow is where one or more topology solutions are created to meet the performance requirements of the design and includes creation and configuration of Ncore caches. Users can manually select and configure each network. 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. 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 Maestro Graphical User Interface.

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 The Maestro Workspace

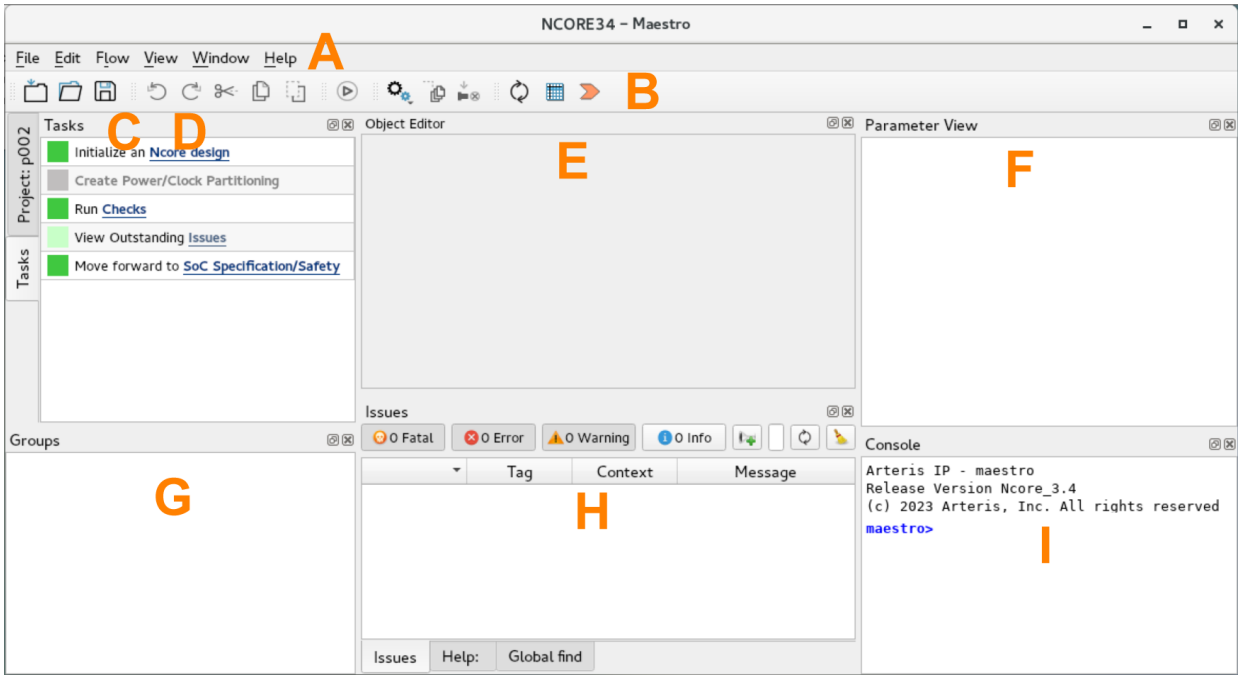


Table 1. The Maestro workspace

Key	Name	Description
A	Maestro Menus	Provides access to Maestro menus. See "Maestro Menus and Shortcuts".
B	Horizontal Toolbar	Provides quick access to Maestro tools and functions. See "Horizontal Toolbar".
C	Tasks View pane	View tasks within a Phase, which provide a workflow for completing the phase. See "Tasks View Pane".
D	Project Tree pane	View the Project Tree of the database. Use the tabs to move between Tasks and Project panes. The Project tab shows the name of the project. See "Project Tree Pane".
E	Object Editor pane	View the information about selected objects. See "Object Editor pane".
F	Parameter View pane	View Maestro parameters. See "Parameter View Pane".
G	Groups pane	Shows project groups.
H	Issues pane	Provides access to Issues, Help, and Global find. See "Console Interface Pane and Other tabs".
I	Console pane	Command line and project log interface. See "Console Interface Pane and Other tabs".
	Dock Widgets	Panes can be undocked and moved using the widget in the upper right of each pane. Use the dock widget to restore the pane. Use X to delete the pane.

2.2 Maestro Menus and Shortcuts

2.2.1 File Menu

The File Menu provides controls and commands for text files.

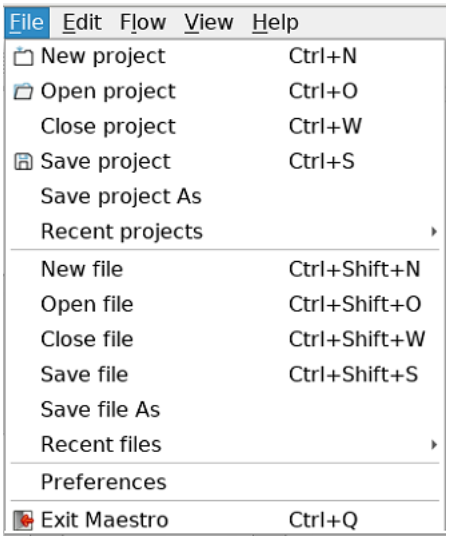


Table 2. File menu descriptions

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.

Table 2. File menu descriptions

Control	Description
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.

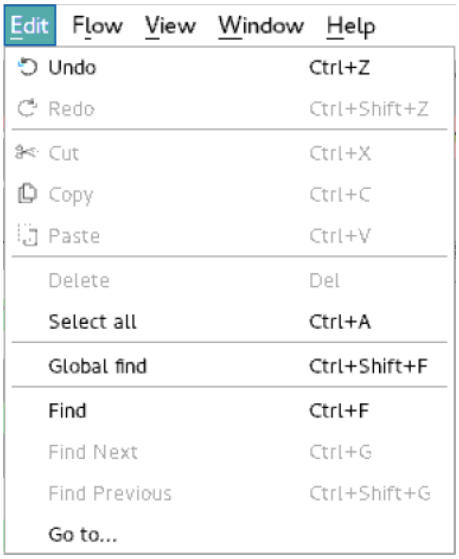


Table 3. Edit menu descriptions

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	Select all objects in a window.

Table 3. Edit menu descriptions

Control	Description
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.

2.2.3 Flow Menu

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

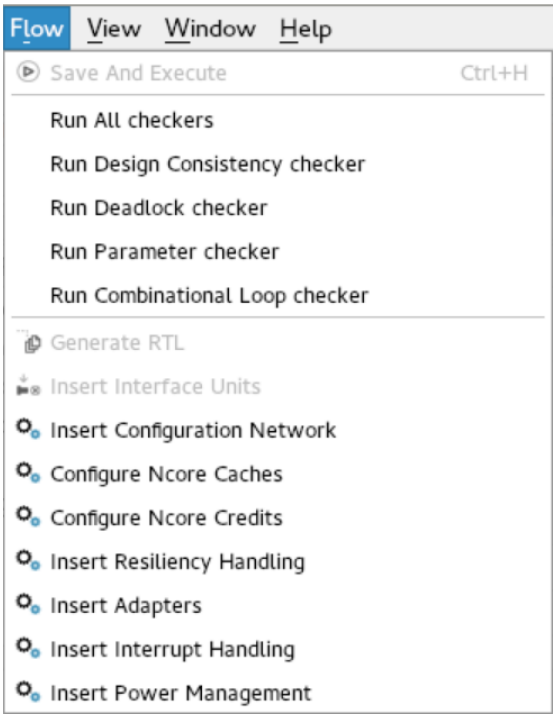


Table 4. Flow menu descriptions

Control	Description
Save and Execute	Save and execute 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 Combinational Loop checker	Run Combinational Loop Checker.

Table 4. Flow menu descriptions

Control	Description
Generate RTL	Generate RTL for the design.
Insert Interface Units	Initialize the solution by inserting the required interface blocks (Ncore units or ATUs) in front of the sockets.
Insert Configuration Network	Inserts a CSR network to configure the components of the solution.
Configure Ncore Caches	Configure the Ncore caches.
Configure Ncore Credits	Configure the Ncore credits.
Insert Resiliency Handling	Inserts the resiliency infrastructure to provide the requested degree of resiliency.
Insert Adapters	Automatically fixes any clock or data width mismatch between connected components by inserting adapters in between.
Insert Interrupt Handling	Inserts the interrupt infrastructure to gather and accumulate interrupt signals.
Insert Power Management	Insert power management.

2.2.4 View Menu

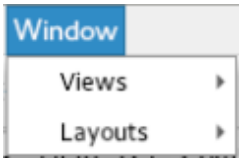
The following table explains the use of controls and icons displayed in the View Menu.

Table 5. View menu descriptions

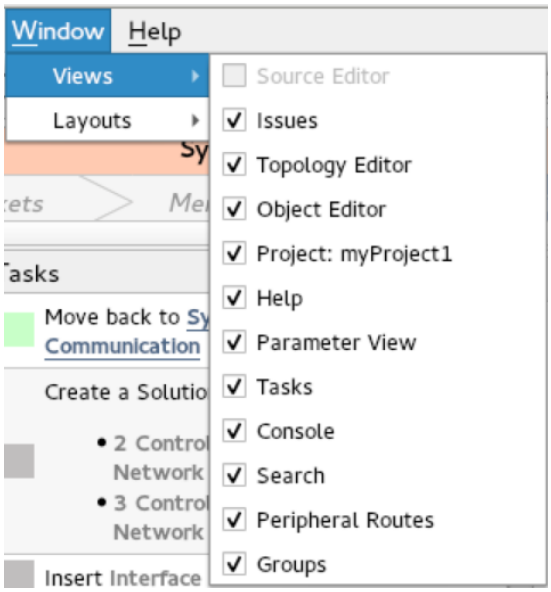
Control	Description
Redraw	Select this icon to force a redraw of the GUI.

2.2.5 Window Menu

The Window Menu has selections for Views and Layouts.



Views has similar selections as the “Maestro Properties Editor” on page 28 and includes selections to view Issues, Topology Editor, Object Editor, Project, Parameters, Tasks, Groups, and so on.



Layouts allows you to return to the Default Layout.

2.2.6 Help Menu

Use the Help Menu to access Online Help (F1), user guide, release notes, technical support information, and so on.

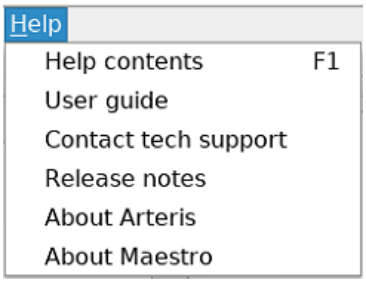


Table 6. Help menu descriptions

Control	Description
Help contents	Open the complete Online Help system.
User guide	Access an Ncore 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.

2.3 Horizontal Toolbar

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

Table 7. 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.
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.
J		Automation	Access sub menus: - Insert Power Management - Insert Interrupt Handling - Insert Resiliency Handling - Insert Configuration Network - Insert Adapters

Table 7. Toolbar Icon descriptions

Key	Icon	Name	Description
K		Generate RTL	Generate RTL for the design.
L		Insert Interface Units	Initialize the solution by inserting the required interface blocks (Ncore units or ATUs) in front of the sockets.
M		Redraw	Force a redraw of the GUI.
N		Toggle Single Table Mode	Toggles to display one folder/object at a time in Object Editor.
O		Toggle Flow Navigator	Toggles the horizontal ribbon which shows the flow phases of the design.

2.3.1 Flow Navigator Ribbon

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

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.

If you edit an object in the Project Tree, the Flow Navigator Ribbon phase state will change to reflect the object that was edited. You will need to manually return to the previous phase state by clicking in the ribbon.

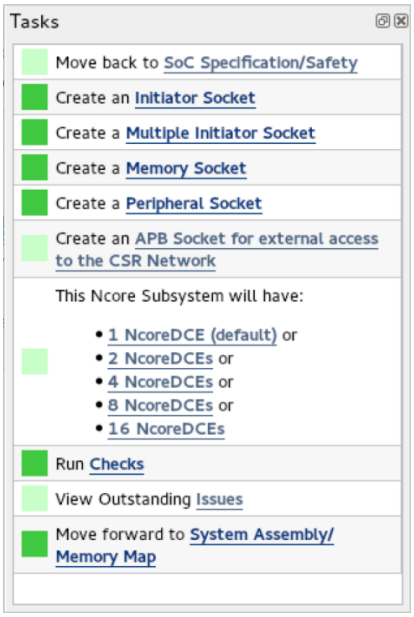
2.4 Maestro Panes

The pains used by Maestro include:

- "Tasks View Pane"
- "Project Tree Pane"
- "Object Editor pane"
- "Parameter View Pane"
- "Console Interface Pane and Other tabs"

2.4.1 Tasks View Pane

The figure below shows the various subsections within the Tasks View Pane of a particular Maestro Phase (System Assembly/Sockets). 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, by clicking “Move back ...” or by using the Flow Navigator ribbon. When you are done with a phase, you run checks and address outstanding issues and then move forward to the next phase.

The Tasks pane uses green and gray squares to indicate task states within each phase, as shown in this example for the SoC Specification phase.

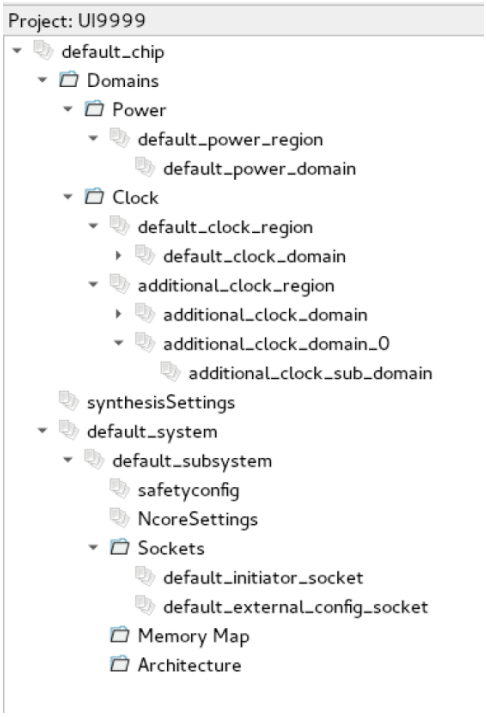
- A green square indicates a task needs to be done.
- A faded green square indicates the task has been performed, but can be performed again (e.g., inserting a socket).
- A gray square indicates a task has been performed and does not need to be done again (e.g., inserting interface handlers), or the task is not applicable in the current configuration.
- 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.

2.4.2 Project Tree Pane

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

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 3. Project Tree Example



2.4.3 Object Editor pane

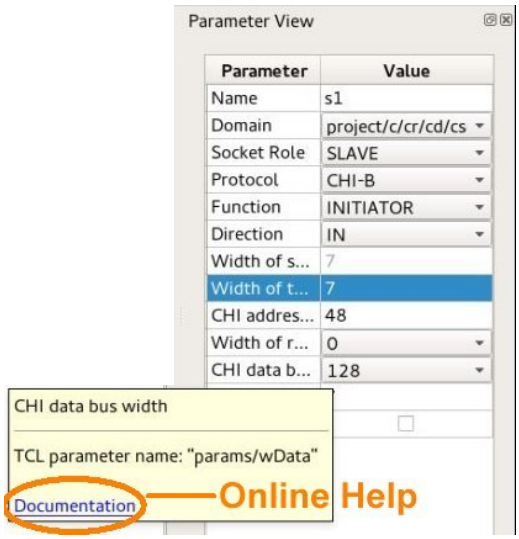
The Object Editor displays parameters, values, and data about selected objects in the middle window of the Project Tree. Select the Object Editor tab to view information or close the Object Editor.

2.4.4 Parameter View 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 and Online Documentation Access

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 Online Help Documentation for the parameter (click the blue link). Note: Parameters are described in *Maestro Parameters*.

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.

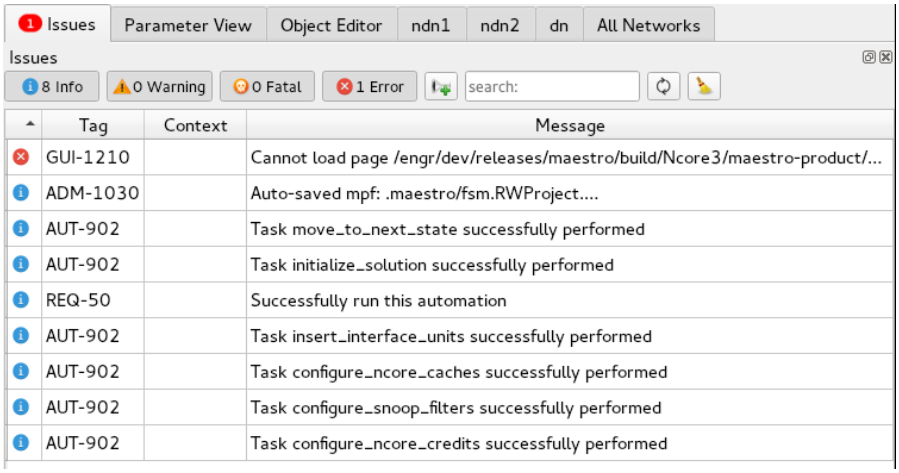
2.4.5 Console Interface Pane and Other tabs

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

Figure 4. Console Interface

- **Console tab** : Shows Maestro actions.

- **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 all. Select one or more checkboxes to set the search parameters.
- **Help tab** : Displays help.
- **Global find tab** : Enter terms to locate an item or other information in the Arteris IP database. The search has filters (case, wildcards) and results are displayed in the console window. The results are formatted with a hyperlink to enable direct access to the selected object.

2.5 Topology Editor Toolbar

The Topology Editor toolbar is visible when using the Topology Editor (Window > Views).

Table 8. 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		Zoom 1:1	Zoom to 1:1 ratio (no zoom in/out).
D		Zoom In	Zoom window in.
E		Zoom Out	Zoom window out.
F		Zoom Selected	Fit view to selection.
G		Fit All	Fit view to all.
H		Relayout	Move from layout to grid view.
I		Stretch to View	Stretch to view in window.
J		Distribute Adapters	Distribute adapters.

Table 8. Topology Toolbar Icon descriptions

Key	Icon	Name	Description
K		Prepend Adapter	Prepend an adapter to selected input.
L		Prepend Switch	Prepend switch to all inputs of the selected object.
M		Append Adapter	Append an adapter to selected output.
N		Append Switch	Append switch to all outputs of the selected object.
O		Insert Adapter	Insert adapter to selected routes.
P		Insert Switch	Insert switch on selected routes.
Q		Split Switch	Split the selected switch.
R		Delete Item	Delete the selected item.
S		Create Unconnected Switch	Create an unconnected switch.
T		Merge Switch	Merge selected switches or toggle mode where switches can be merged by drag and drop.
U		Toggle Select Routes Mode	Toggle mode where routes are selected by clicking links.
V		Toggle Topology View	Toggle topology view configuration pane visibility.

2.6 Additional UI Features

2.6.1 Undock and restore the Maestro interface

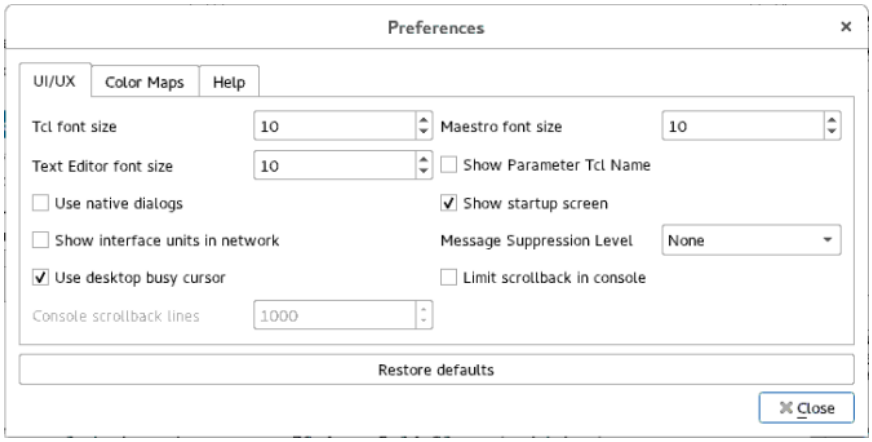
Use these steps to undock and restore the Maestro interface.

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.

2.6.2 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.



The fields on the UI/UX tab can be used to:

- Change the font size for the Tcl console, Text Editor, and Maestro.
- Show parameter Tcl name and startup screen.
- Use native dialogs.
- Show interface units in network.
- Set message suppression level.
- Limit scrollback in the console.

You can also:

- Select the Restore defaults 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.

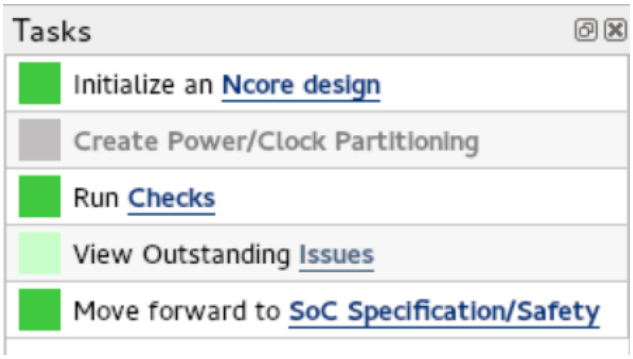
3 SoC Specification (Phase 1)

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.

Tasks for this phase:



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. Ncore 3 only supports one power region and one power domain.

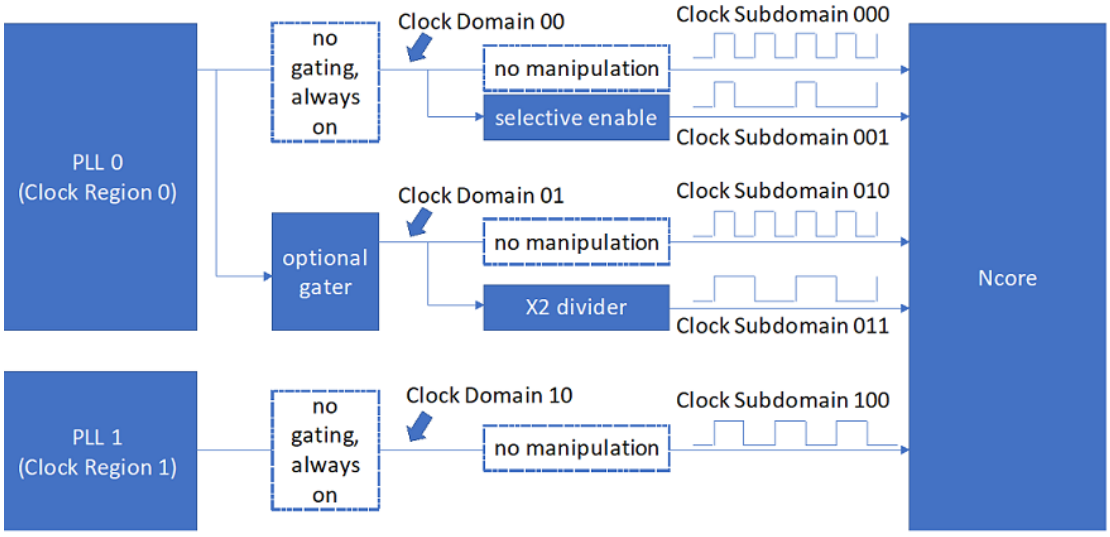
3.1.3 Clock Regions, Clock Domains, and Clock Subdomains

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 5. Clock Regions, Domains, Subdomains, and clock signals



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

A clock domain can only have one clock subdomain. 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. A splash screen is displayed by default. The startup screen can be turned off using the Maestro Properties Editor (File > Preferences).
2. The Flow Navigator 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 -- **s**elect 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. In Tasks View, click “Initialize an Ncore design”.
2. The Project Tree will be populated with a default chip, system, and subsystem (power domain, one clock region, and one clock domain).
3. The name of the chip is called "default_chip". To rename it, double-click on "default_chip" and enter a new name.
4. Click in Tasks View to create additional clock regions and assign clock domains to them.

Use Tasks View as your primary guide as you move through the phases of the design.

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.

3.1.9 Create a Power Domain

This is done automatically during Create Power/Clock Partitioning. A Power Domain consists of one Power Region and one Power Domain.

3.1.10 Configure the Power Region Parameters

To configure Power Region parameters:

1. Click on the Power Region name.
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.
2. The Object Editor will display the parameters in the main window.

3. Power domain Gating is set to “always_on” and cannot be changed.

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. Powering down a Power Domain is not supported.

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 (clock domain). If different gating signals can modulate the clock from the same clock region, each must be modeled as a separate clock domain. Ungated propagation is also modeled as its own clock domain. Since powering down is not supported, the gating option is always “always_on”.

3.1.12 Create a Clock Domain

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

3.1.13 Configure the Clock Region Parameters

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

1. 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. In the Object editor or parameter view, 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

The only setting is “always_on”.

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. Divide by parameters for clock sub domains are not supported.

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.

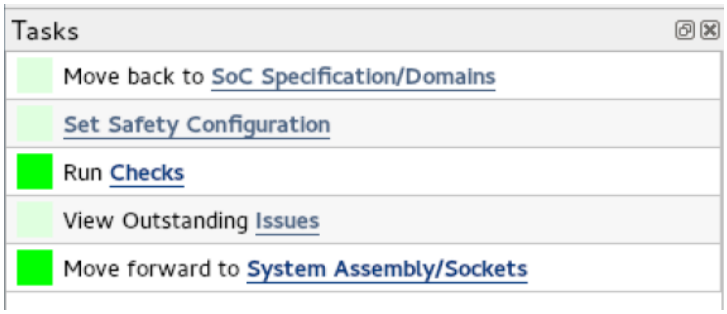
Select the Project Tree and complete the following steps.

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.

3.2.2 Setting the Safety Configuration

After running checks in SoC Specification/Domains, the next step is to move forward to SoC Specification/Safety.

Tasks for this phase:



Click “Set Safety Configuration” to select resiliency and functional safety setting levels. Safety settings in Ncore can be set for designs in which those features are required (e.g., automotive chips). Ncore 3 supports 3 different ASIL levels: ASIL_A, ASIL_B, and ASIL_D.

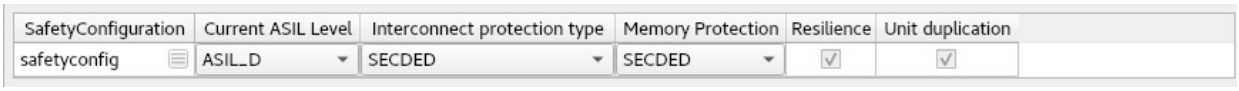
Following are the safety configurations which can be chosen. If the value is listed as user settable, then the value picked by the ASIL setting can be later changed by the user.

- NO_ASIL : resilienceEnabled is false, duplicationEnabled is false, memoryProtectionType is NONE and user-settable, resiliencyProtectionType is NONE and user-settable.
- ASIL_A : resilienceEnabled is true, duplicationEnabled is false, memoryProtectionType is PARITY and user-settable, resiliencyProtectionType PARITY is user-settable.
- ASIL_B : resilienceEnabled is true, duplicationEnabled is true, memoryProtectionType is PARITY and user-settable, resiliencyProtectionType PARITY is user-settable.
- ASIL_D : resilienceEnabled is true, duplicationEnabled is true, memoryProtectionType is SECDED only, resiliencyProtectionType is SECDED only.
- NOT_SET : No ASIL set.

Note If no safety features are required, then select NO_ASIL.

Values that are mandatory for that ASIL value are not allowed to be changed, so for ASIL-D, all the values are hard coded and frozen to the highest safety settings.

Safety Configuration parameters can be edited in the Object Editor or by selection from the Project Tree. The following example shows the ASIL_D selection.



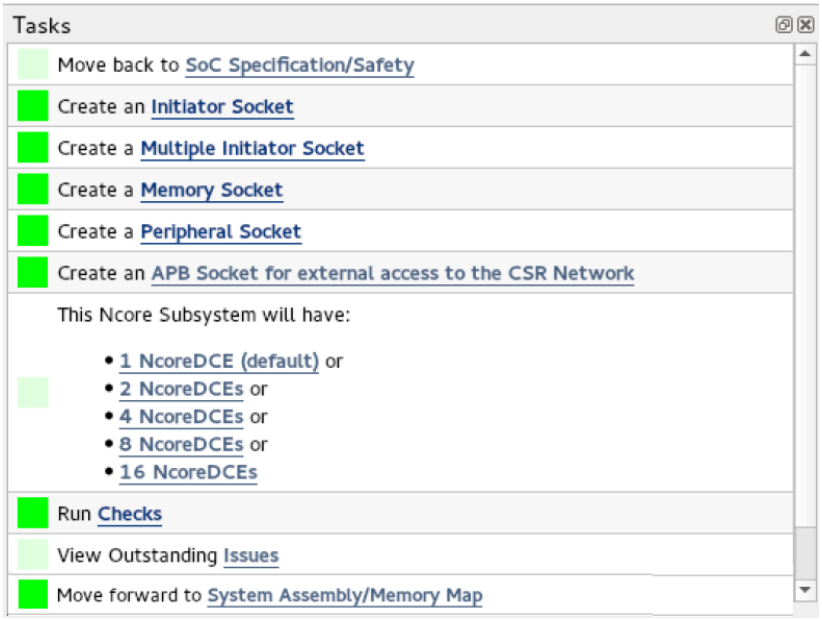
4 System Assembly (Phase 2)

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.

Tasks for 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. Note: An optional APB socket can be created for external access to the CSR network.
- 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 “Common Socket Parameter Descriptions” on page 44.

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.

Multiple Initiator Sockets can be used for more efficient network connectivity by using sockets with multiple ports which share an SMI interface. This allows integration of more IP without having more network connectivity (see “Multiple Initiator Socket” on page 45).

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 6 shows sockets distributed around the edge of the Ncore subsystem.

Table 9 shows the Cache-Coherent Interconnect Socket NIU bindings and supported protocols.

Figure 6. Subsystem and Socket Example

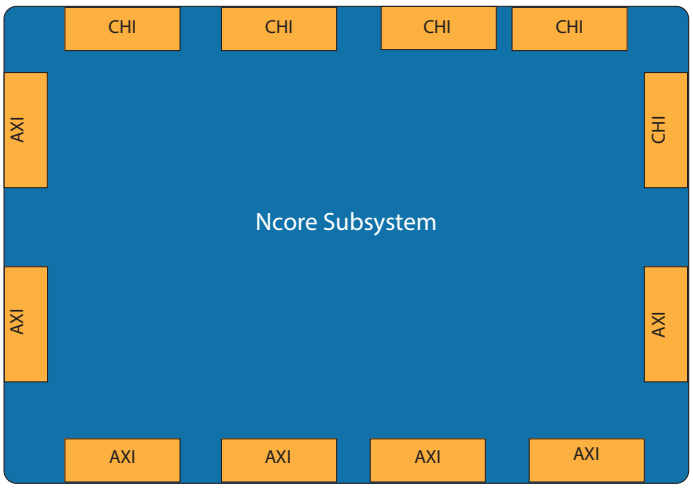


Table 9. Cache-Coherent Interconnect Socket NIU bindings and supported protocols

Interface Unit Name	Binds to socket objects and supported protocols
Coherent-Agent Interface Unit (CAIU)	ACE CHI-A, CHI-B
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 64, 128, and 256 bits.

- Width of Address: Specify the address width: The address width options are 32, 40, 44, or 48.
- 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 Multiple Initiator Socket

A Multiple Initiator Socket is a way to create multiple identical NCAIUs and allow them to share one SMI interface. This allows integration of additional IP without having to add additional network connectivity.

- A Multiple Initiator Socket can have either 2, 4, or 8 NCAIUs sharing a single SMI interface.
- Multiple Initiator Sockets can have protocols of type AXI4, ACE-Lite, or ACE-Lite-E.

4.2.4 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.

4.2.5 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.

4.2.6 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 AXI data bus width parameter value.

4.2.7 Create an APB Socket for External Access to the CSR Network

An APB socket must be created to allow alternative access to the CSR Debug and Trace registers in case of a system hang.

4.2.8 Selection of Number of DCEs

The user can select between 1 and 16 DCEs for their system. Each DCE is configured with identical snoop filters.

4.2.9 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".

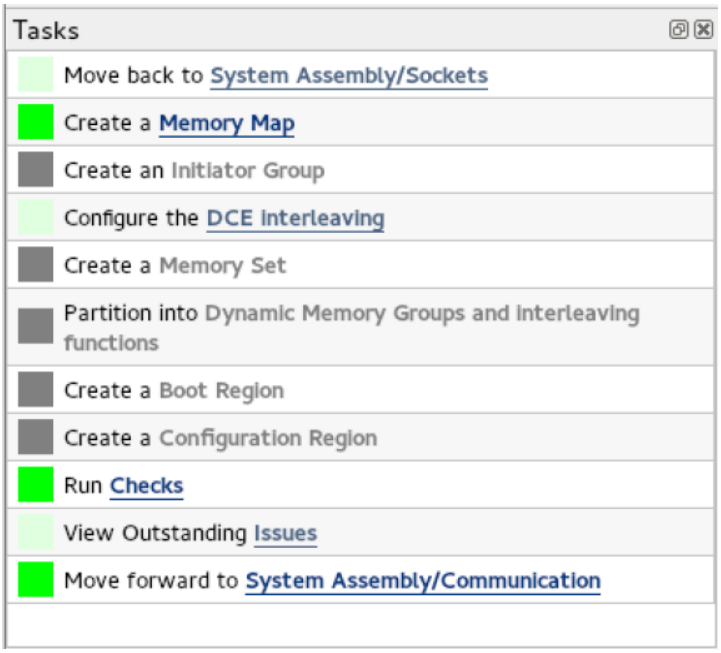
4.3 Memory Map

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

Before you begin

Read the sections below 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.

Tasks for this phase:



4.4 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 16 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. You can have 16 DMIs in 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 MemoryGroup01
  - DMI4 in MemoryGroup02
- MemorySet 1 contains 2 groups — MemoryGroup10 and MemoryGroup11:
  - DMI0, DMI1, DMI2, DMI3 in MemoryGroup10
  - DMI4 in MemoryGroup11

4.4.1 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:

- MemoryGroup0 of MemorySet0 uses a two-way interleaving function because there are only two DMIs: DMI0 and DMI1. MemoryGroup0 of MemorySet0 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.

4.4.2 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.

4.4.3 Create Initiator Groups

Initiator Groups are groups of identical initiator sockets. Initiator groups allow initiator sockets to be interleaved just like memory maps allow DMIs to be interleaved.

DMIs, DCEs, and Initiator Groups can all be interleaved. If all of these components are interleaved using the same bits, the amount of routing connectivity in the topology can be reduced dramatically.

As an example, assume we have an initiator group of 4 initiators and a memory map of 4 DMIs and 4 DCEs, which are all interleaved using bits 8 and 9. In this configuration each Initiator would be connected to one DMI and one DCE. Without interleaving every initiator would need to connect to every DMI and every DCE. Interleaving allows large coherent NoCs to still be implementable.

4.4.4 DCE Interleaving

DCE interleaving functions similar to interleaving for initiator groups and for DMIs. By specifying common interleaving bits between the various components, Ncore is able to dramatically reduce the connectivity of the coherent NoC.

4.4.5 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.

4.4.6 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.

4.4.7 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 (LSB)

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. If there are 16 DMI, you might need to have 4 interleaving bits.

4.4.8 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.

4.4.9 Create Configuration Region

To create a Configuration Region, complete the following steps.

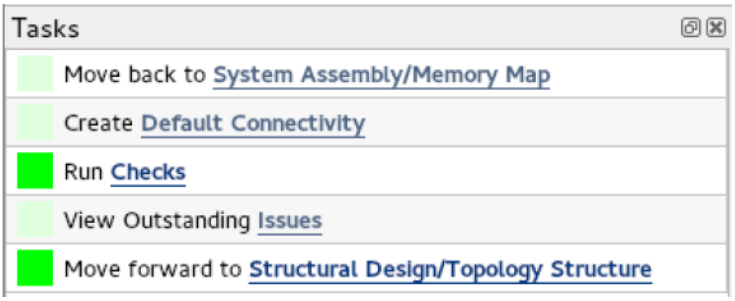
1. In the Project Tree, right-click on a Memory Map. The Create > MemoryMap Region page is displayed.
2. Enter a name for the new Configuration Region in the text field.
3. Select OK. The name of the new Configuration Region is displayed in the Configuration Region directory in the Project Tree.

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.
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 System Assembly/ Communication phase.

4.5 System Assembly/Communication

Tasks for this phase:



4.5.1 Create Default Connectivity

Creating default connectivity hooks up all initiator sockets to the peripheral sockets. A grid is then displayed showing this connectivity map, similar to a chess board.The Connectivity Editor allows for the modification of, or the removal of, connections. You also have to ability to prune connectivity between initiators and DII to allow for more efficient designs.

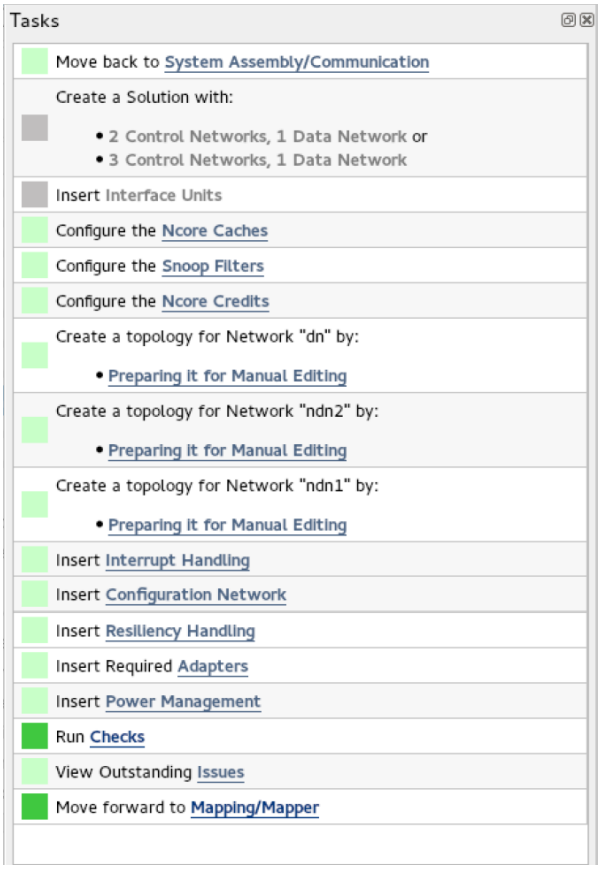
5

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
  - Insert Configuration Network
  - Insert Resiliency
- Use the Mapping tool in the Flow Navigation tab to view the network designs.

Tasks for this phase:



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
- 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. A transport solution will always have one data network, but it can have either two or three control networks. Increasing the number of control networks allows for more parallelism.

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

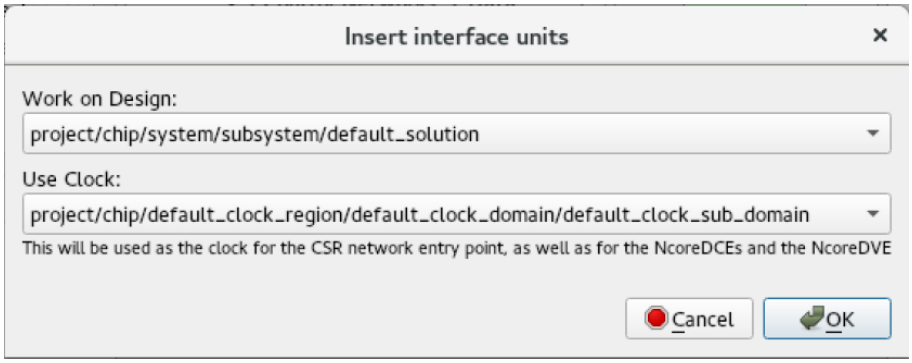
After a template is selected, the Architecture directory is automatically updated with a new directory named Networks. The directory contains subdirectories for each control network and for the data network. A directory for Interface Units is also added to the Networks directory.

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 the topology.
2. Select the Insert Interface Unit icon in the toolbar.



3. Verify and change the settings, if required. The settings are described below.

- Work on Design: Refers to the solution created under the subsystem.
- Use Clock: Refers to the clock subdomain created in Phase 1.

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.

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 Configure Ncore Caches

Ncore caches store coherent copies of data and implement snooping interfaces. The Ncore system can be configured with two different types of caches. The optional cache configured within an NCAIU is referred to as proxy cache, and the optional cache configured within a DMI is referred to as system memory cache. In

this form the user selects which of the NCAIUs will be configured with a proxy cache and which DMIs will have a system memory cache.

5.2.4 Configure Snoop Filters

Ncore uses snoop filters to reduce the system bandwidth consumed to enforce coherence by eliminating unnecessary coherency operations. The system may be configured with one or more snoop filters, and each caching agent must be assigned to one and only one snoop filter. As a result, each snoop filter can be configured to match the caching characteristics of the associated caching agent.

In the snoop filter form the user selects the number of snoop filters the design has, their number of sets and ways, their configuration bits and which caching agents are assigned to each of them.

5.2.5 Configure Ncore Credits

You can configure the Ncore Credits for the design. In the form, all the devices with credits will be highlighted. Update the credits as needed to achieve the desired performance.

5.2.6 Create a Topology : Preparing it for Manual Editing

When you prepare a topology for manual editing, Ncore creates a starting topology for you. In this topology every Ncore unit is attached to one switch, by default. These initial switches cannot be deleted but they can be merged together with other switches.

Use the Topology Editor tools 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.

5.2.7 Topology Editor : View the Structural Design Architecture

The Topology Editor displays the network design and displays what has been created.

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.

5.2.8 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.

5.2.9 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.

5.2.10 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. The Resiliency parameter is visible in the Safety Config object under the Subsystem object.
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.

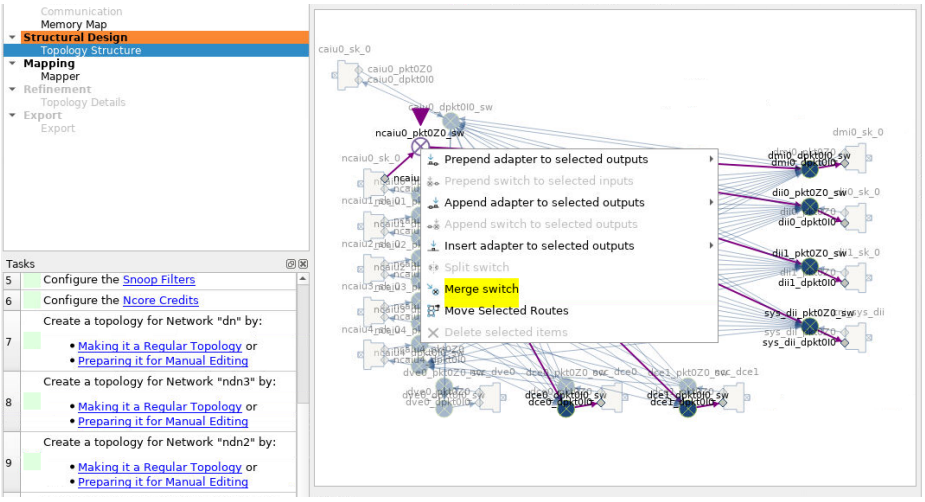
6. Select OK to run the Insert Resiliency tool.

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

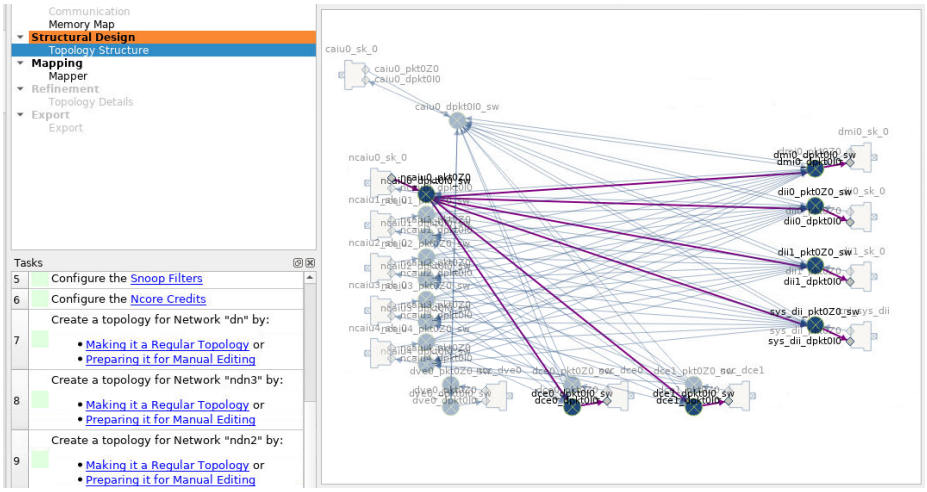
5.3 Merging Two Switches

Use these steps to merge two switches into one switch.

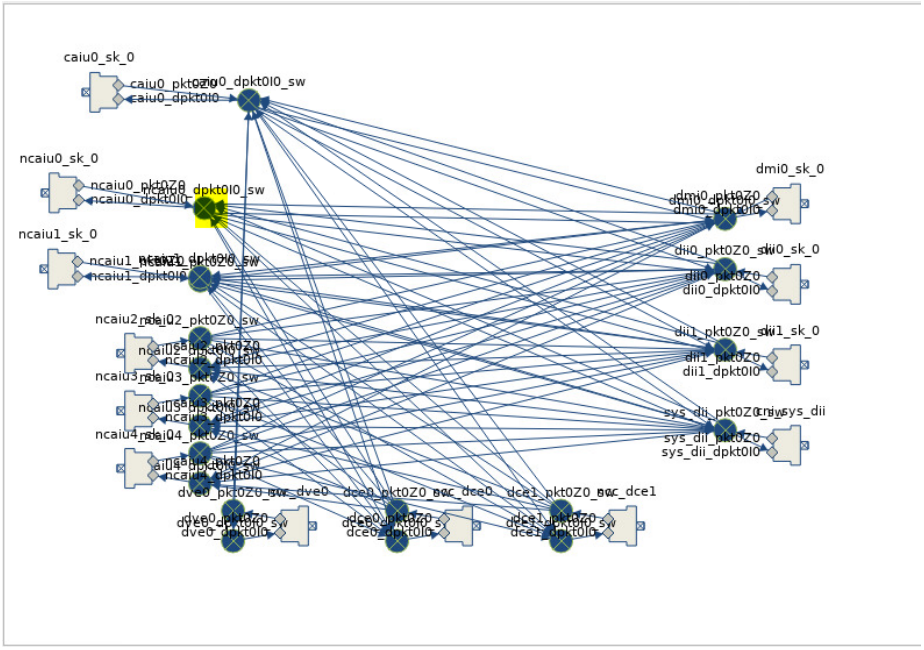
1. In the Topology editor, left-click on a switch and use Ctrl + left-click on all the switches you desire to merge.
2. You can then either use the toolbar Merge switch command, or right-click and select the Merge switch.



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



4. The two switches are merged into one switch.



Other example exercises (prepending/appending/inserting adapters and switches, and so on) can be found in “Topology Editing Exercise” on page 88.

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

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 26 for details on the toolbar.

7.2.2 Parameter View

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

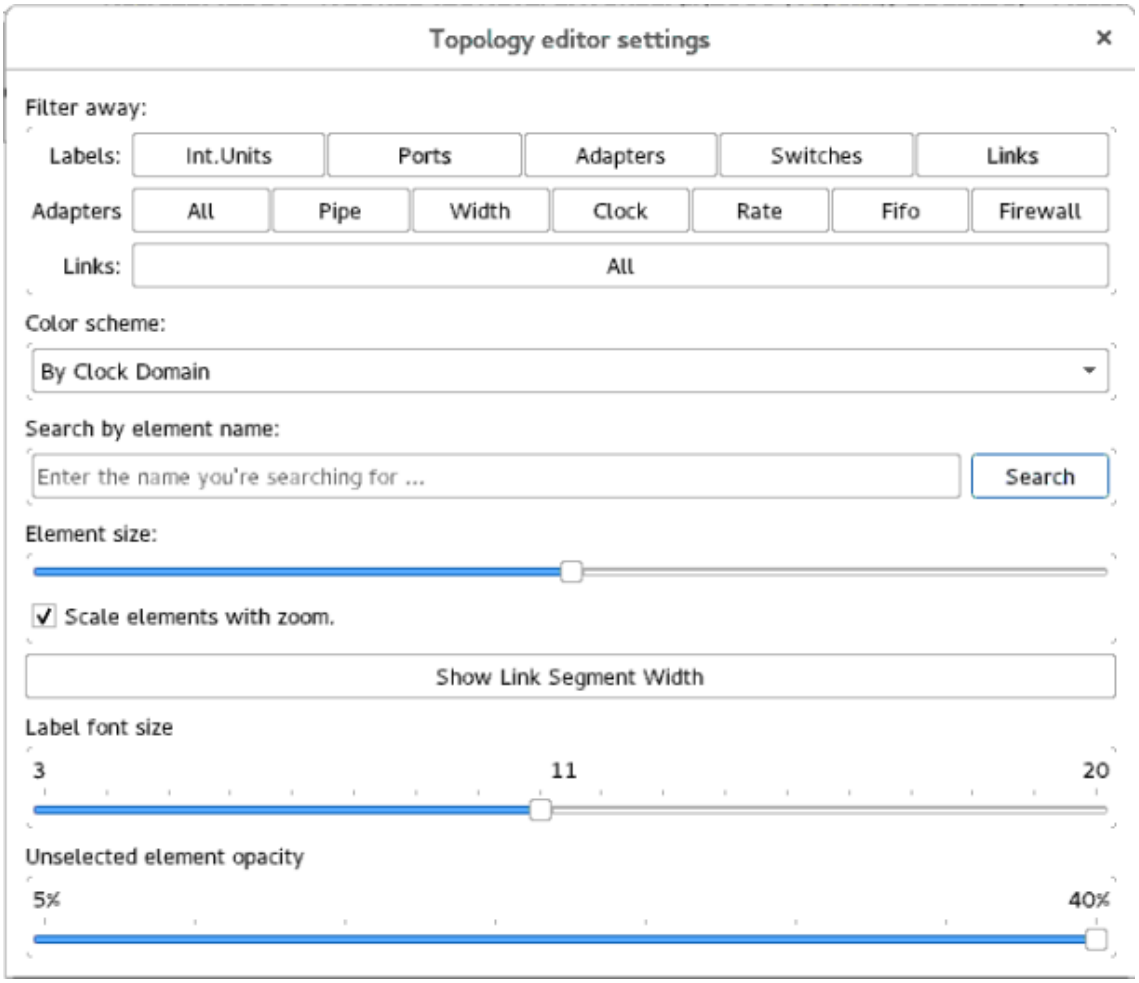
You can change the values for the selected object in the Parameter View.

7.2.3 Topology Filters Editor

You can use the Filters editor to control the appearance and presentation of objects in a network topology editor. Changes made in the Filters editor are applied immediately to all network diagrams.

For example, if you select the All button, then all adapters are hidden in all network diagrams.

Figure 7. 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. This will put the system into Select Routes mode. You can exit Select Routes mode using the Select Routes button in the toolbar.

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 workflow for this demonstration design 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"

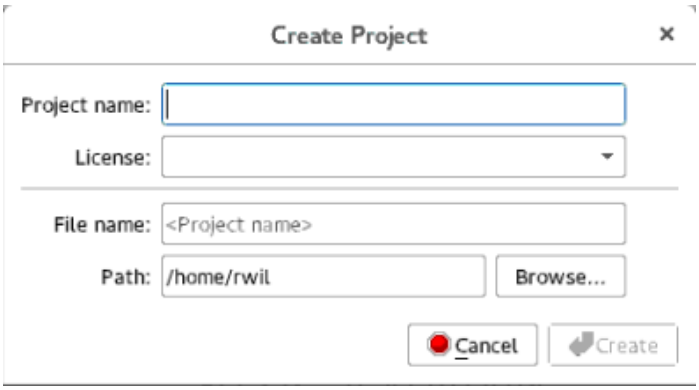
A topology exercise uses the design above to show editing techniques:

- "Topology Editing Exercise"

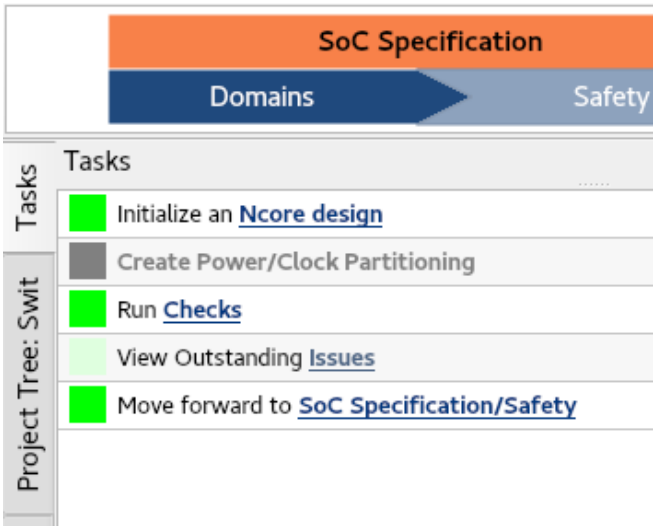
9.2 Phase 1 - SoC (Chip) Specification

Goals for this phase: Domains, Safety (refer to "SoC Specification (Phase 1)" on page 31).

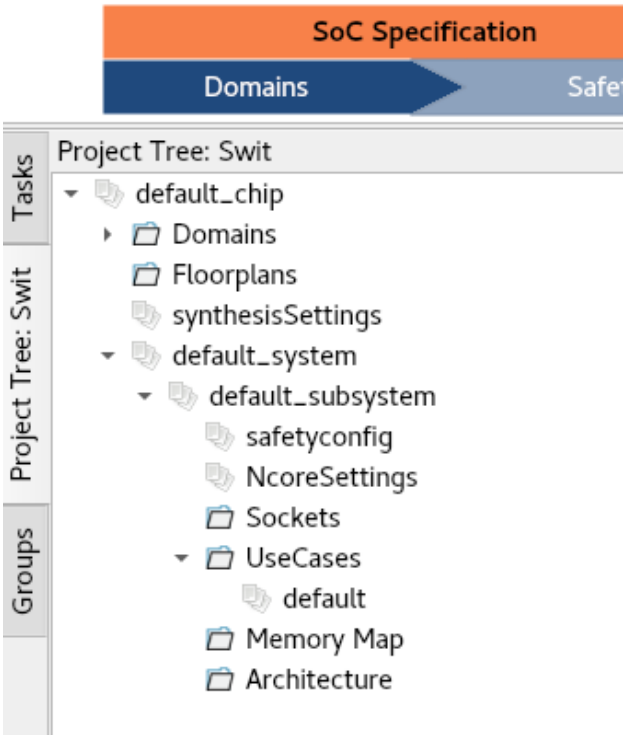
1. Launch Maestro.
2. Click "**New Project**".



3. Enter Project information, select license, and click “**Create**”.

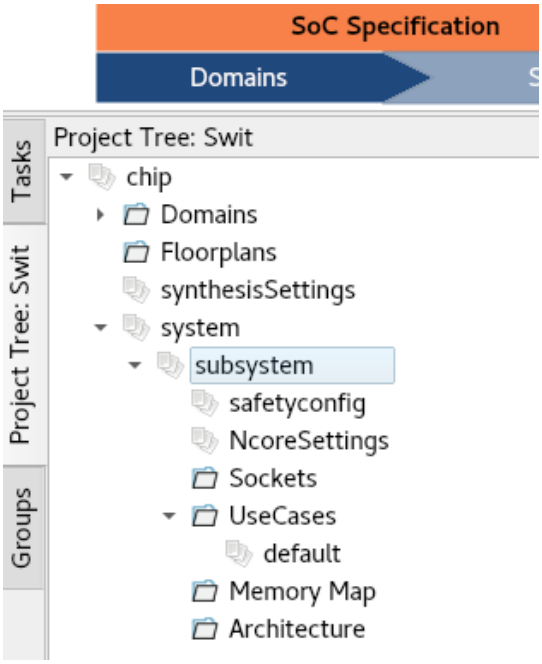


4. From the *Tasks list* select **Initialize an Ncore design**.

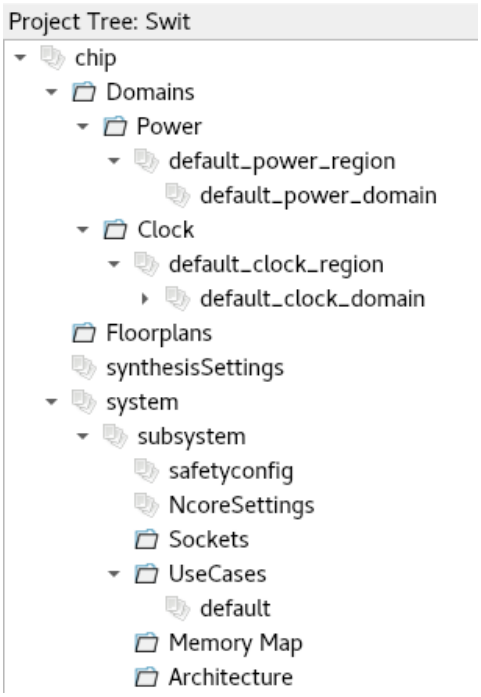


5. Use the *Project Tree* to:
   a. Rename default_chip to “chip”.
   a. Rename default_system to “system”.
   a. Rename default_subsystem to “subsystem”.

6. Double-click on the Project Tree names to enable editing.

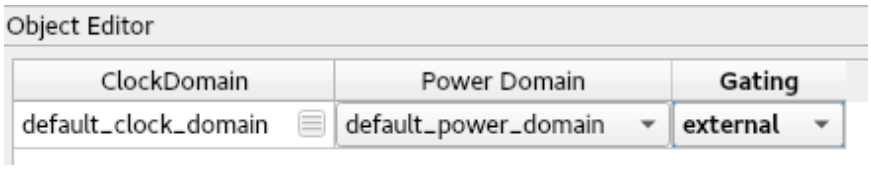


7. Use the *Tasks list* to **Create Power/Clock Partitioning**.



8. Use the *Object editor* to view **Gating** for the **default_power_domain.** Gating is set to “always_on” and cannot be changed.

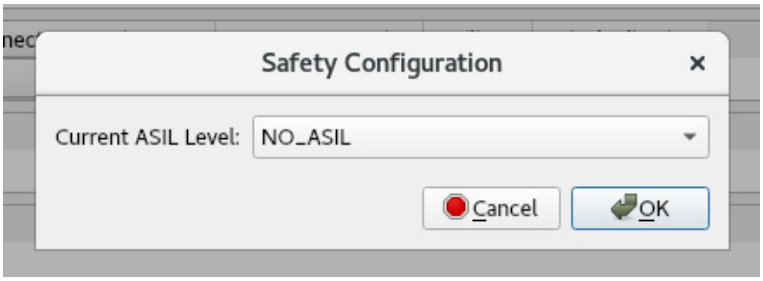
9. Use the *Object editor* to set **Gating** to *external* for the **default_clock_domain.**



10. Use the *Tasks list*, Run Checks and View Outstanding Issues. Note: It is good practice to run checks and make sure all issues have been addressed before continuing to the next phase. You should also save the design before proceeding to the next phase.

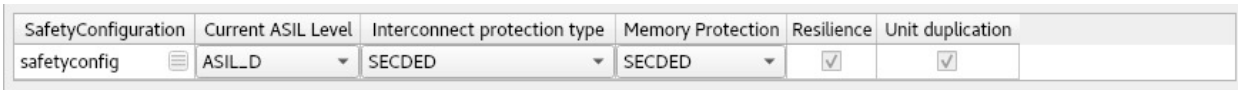
11. Use the *Tasks list* to navigate to **SoC Specification/Safety**.

12. To Select resilience and functional safety settings, select the *Set Safety Configuration* options in *Tasks list*.



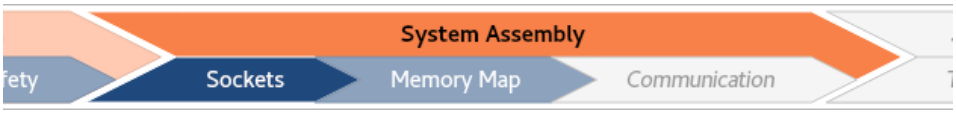
13. The Safety Configuration dialog should show up.

14. Select the **ASIL_D** level.



15. Parameters in safetyconfig object should be derived as follows: Row should show up in object editor view, else it can be selected from the Project Tree.

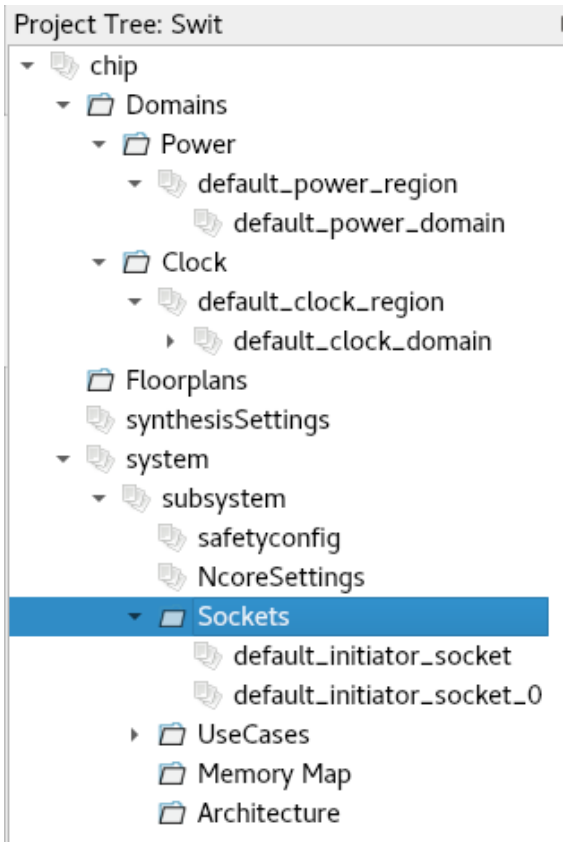
16. Use the *Tasks list* to navigate to **System Assembly/Sockets**.



9.3 Phase 2 - System Assembly

Goals for this phase: Sockets, Memory Map, Communication (refer to “System Assembly (Phase 2)” on page 41).

1. Use the *Tasks list* to create **two Initiator Sockets** (click twice).



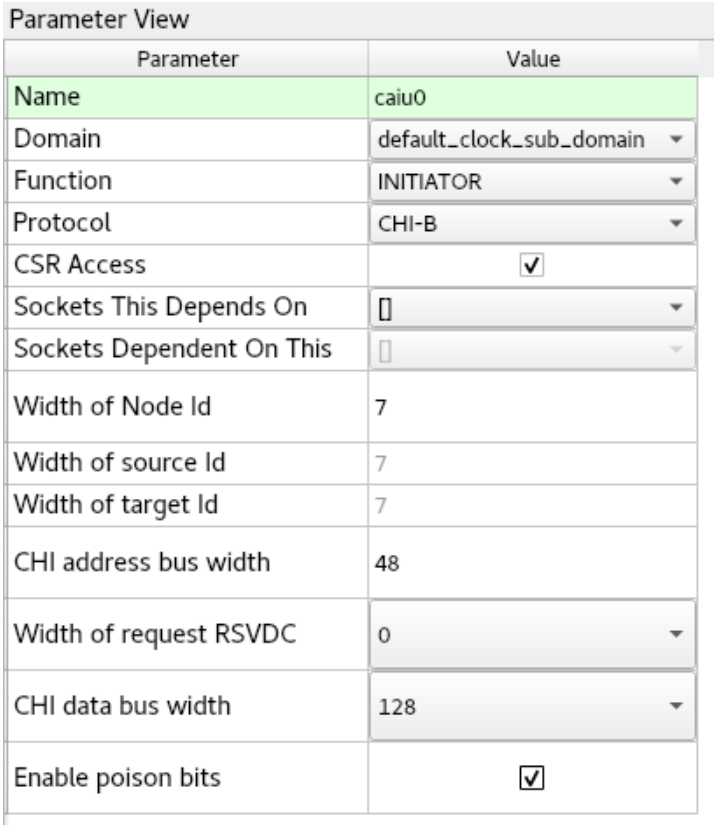
2. In the *Project Tree* select "**default_initiator_socket**".

Parameter View

Parameter	Value
Name	default_initiator_socket
Domain	default_clock_sub_domain
Function	INITIATOR
Protocol	CHI-B
CSR Access	☑
Sockets This Depends On	[]
Sockets Dependent On This	[]
Width of Node Id	7
Width of source Id	7
Width of target Id	7
CHI address bus width	48
Width of request RSVDC	0
CHI data bus width	128
Enable poison bits	☐

3. In the *Parameter View* update **default_initiator_socket**.
4. Change *name* to "**caiuo**".

5. Toggle *Enable poison bits* **on.**



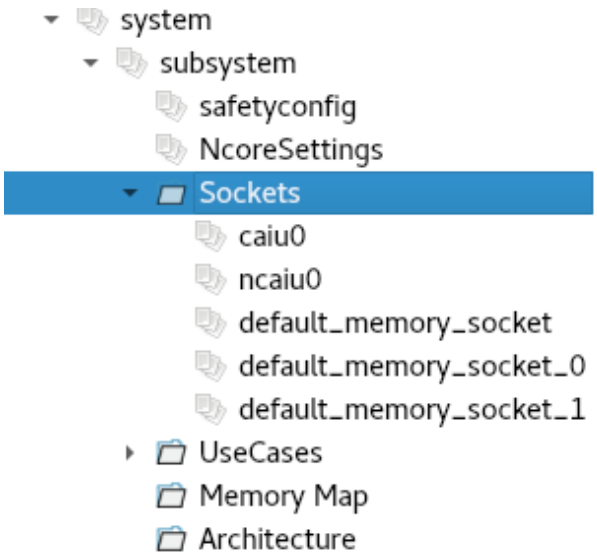
6. In the *Project Tree* select "**default_initiator_socket_0**".
   The **default_initiator_socket_0** is displayed in the Parameter View.
7. In the *Parameter View* update **default_initiator_socket_0.**
   a. Change *name* to "ncaiu0".
   b. Change the *Protocol* to **AXI4.**

c. Change *AXI address bus width* to **48**.

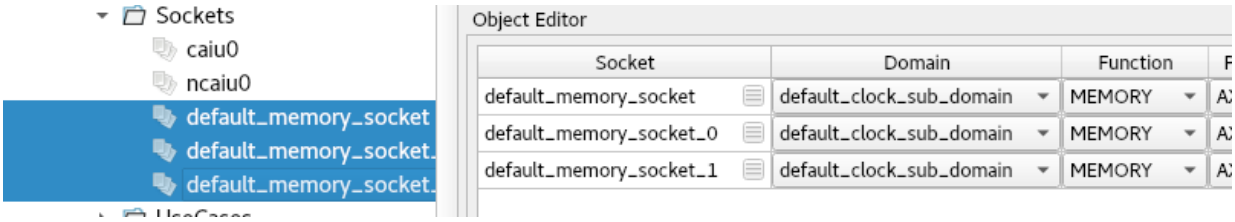
Parameter View

Parameter	Value
Name	ncaiu0
Domain	default_clock_sub_domain
Function	INITIATOR
Protocol	AXI4
CSR Access	☑
Disable Read Interleave	☐
Sockets This Depends On	[]
Sockets Dependent On ...	[]
Width of Ar Id	6
Width of Aw Id	6
AXI address bus width	48
AXI data bus width	64
Width of Aw user	0
Width of W user	0
Width of B user	0
Width of Ar user	0
Width of R user	0
Width of size	3
Width of region	0

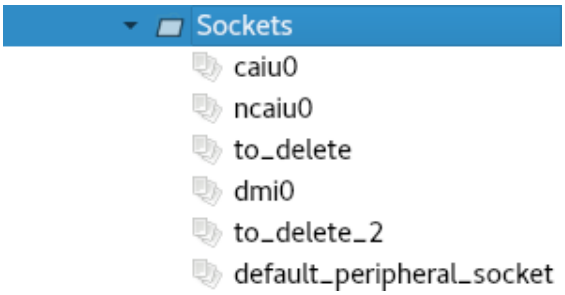
8. Use the *Tasks list* to create 3 **Memory Sockets** (click 3 times)**.**



9. In the *Project Tree* select **default_memory_socket, default_memory_socket_0,** and default_memory_socket_1.



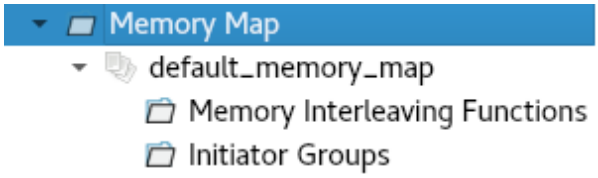
10. Using the Object Editor:
    a. Rename **default_memory_socket** to "**to_delete**".
    b. Rename **default_memory_socket_0** to "**dmio" .**
    c. Rename **default_memory_socket_1** to "**to_delete_2**". We will delete to_delete and to_delete_2 in a later step.
11. The Object Editor displays the updated values.
12. Use the *Tasks list* to create a *Peripheral Socket*.



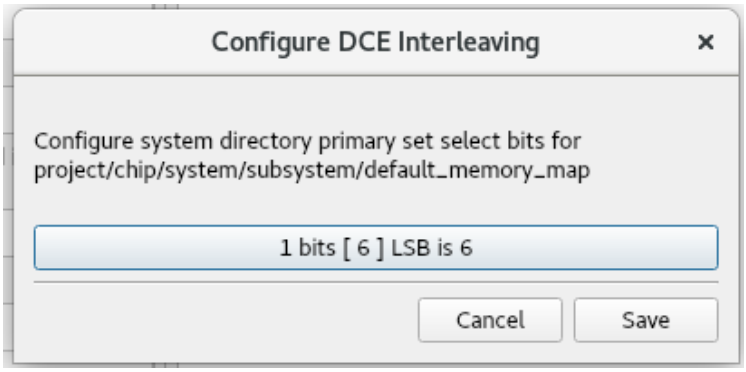
13. Use the *Project Tree* to rename **default_peripheral_socket** to **diio.**
14. The Project Tree displays the updated values.
15. Use *Tasks list* to select "**This Ncore Subsystem will have 2 NcoreDCEs**".

16. Use the *Tasks list* to navigate to **System Assembly/Memory Map**.

17. Use the *Tasks list* to create a **Memory Map.**



18. Use the *Tasks list* to **Configure the DCE interleaving**.

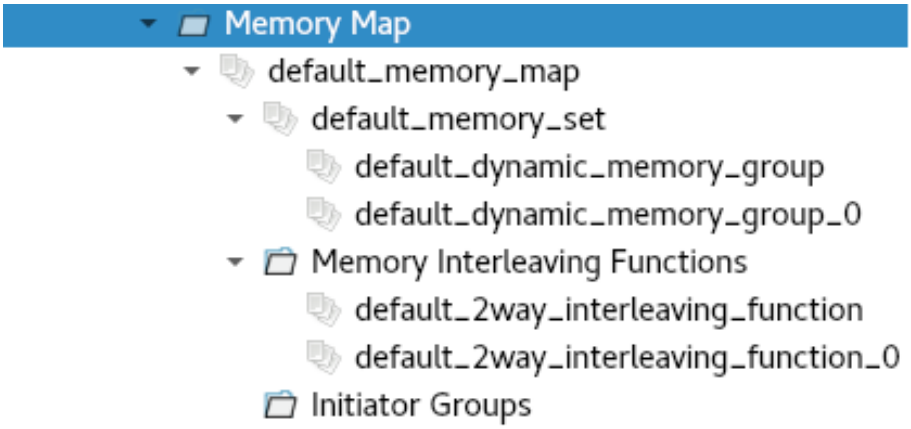


19. Click **Save** to *configure DCE interleaving*.

Parameter View

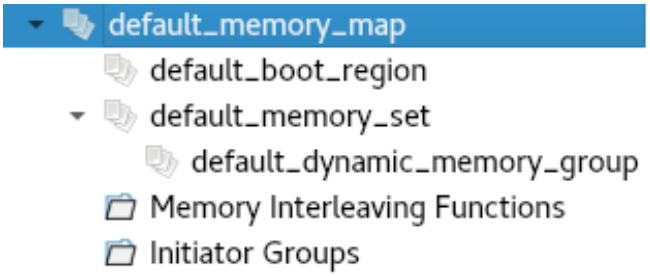
Parameter	Value
Name	default_memory_...
System Directory primary set select bits	[6]

20. Use the *Tasks list* to create a **Memory Set**.
A **memory set** is created.
21. Use the *Tasks list* to **Partition MemorySet "default_memory_set".**

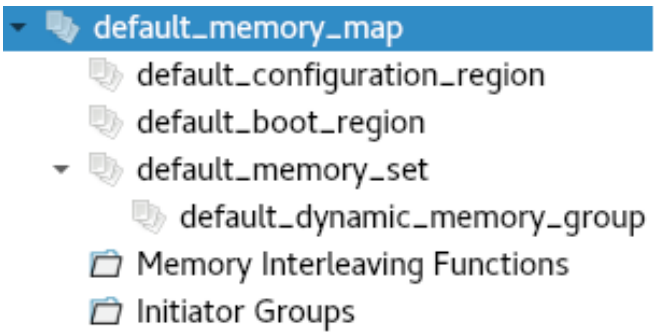


22. Use the *Tasks list* to navigate back to **System Assembly/Sockets.**
23. The project enters the **System Assembly/Sockets** stage.
24. Use the *Object Editor* to delete **Sockets to_delete** and **to_delete_2**.
The sockets are removed from the project.
25. **Undo twice** to recover the missing sockets.
The sockets are reintroduced to the project
26. **Redo twice** to remove the sockets.
The sockets are removed from the project.
27. Use the *Tasks list* to navigate to **System Assembly/Memory Map.**
28. The project enters **System Assembly/Memory Map.**

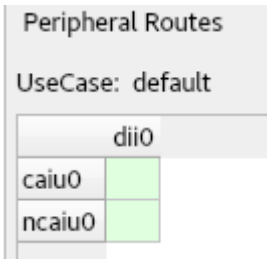
29. Use the *Tasks list* to create a **Boot Region.**



30. Use the *Tasks list* to create a **Configuration Region**.



31. Use the *Tasks list* to navigate to **System Assembly/Communication**.
32. The project enters **System Assembly/Communication**.

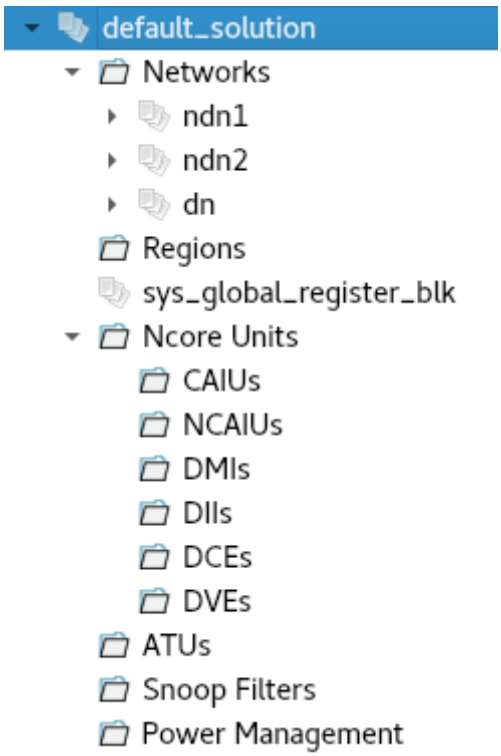


33. Use the *Tasks list* to navigate to **Structural Design/Topology Structure**.

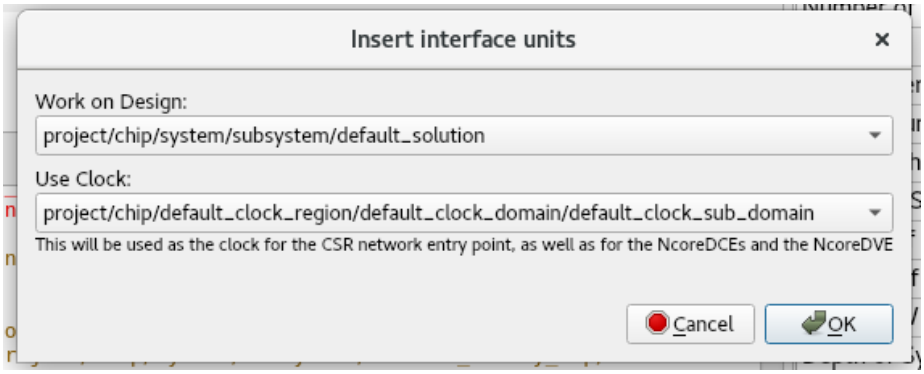
9.4 Phase 3 - Structural Design (Architecture)

Goals for this phase: Topology Structure (refer to “Structural Design (Architecture, Phase 3)” on page 53.

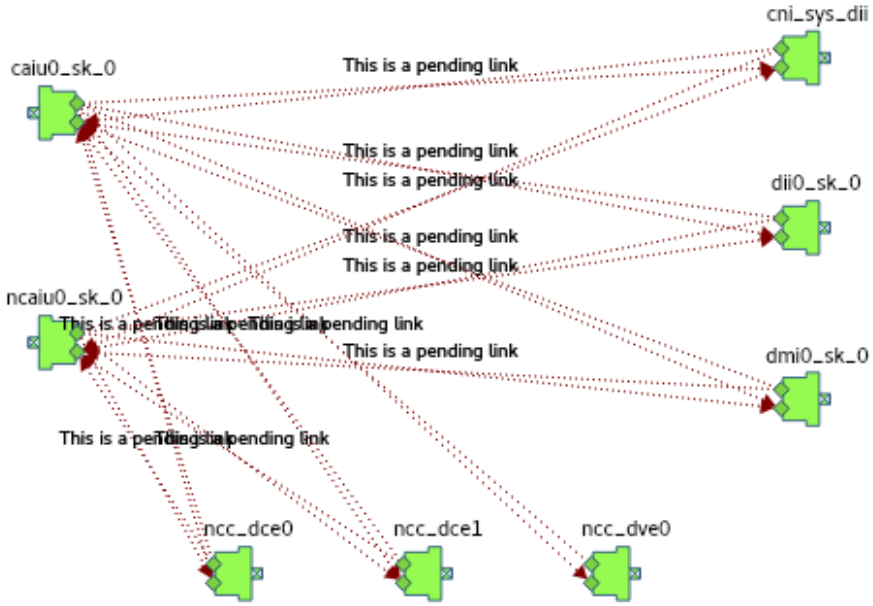
1. Use the *Tasks list* to **Create a Solution with 2 Control Networks and 1 Data Network.**



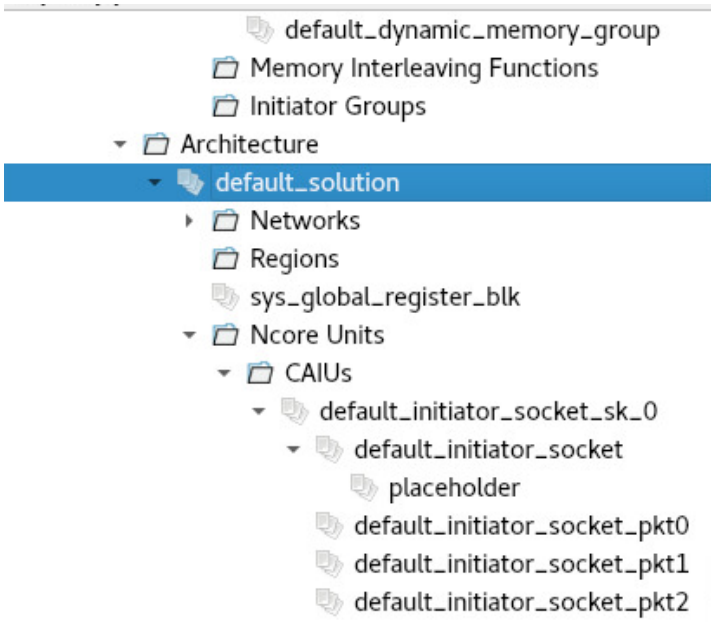
2. Use the *Tasks list* to **Insert Interface Units.**



3. Use the **Run Interface Inserter dialog** to run the interface inserter.

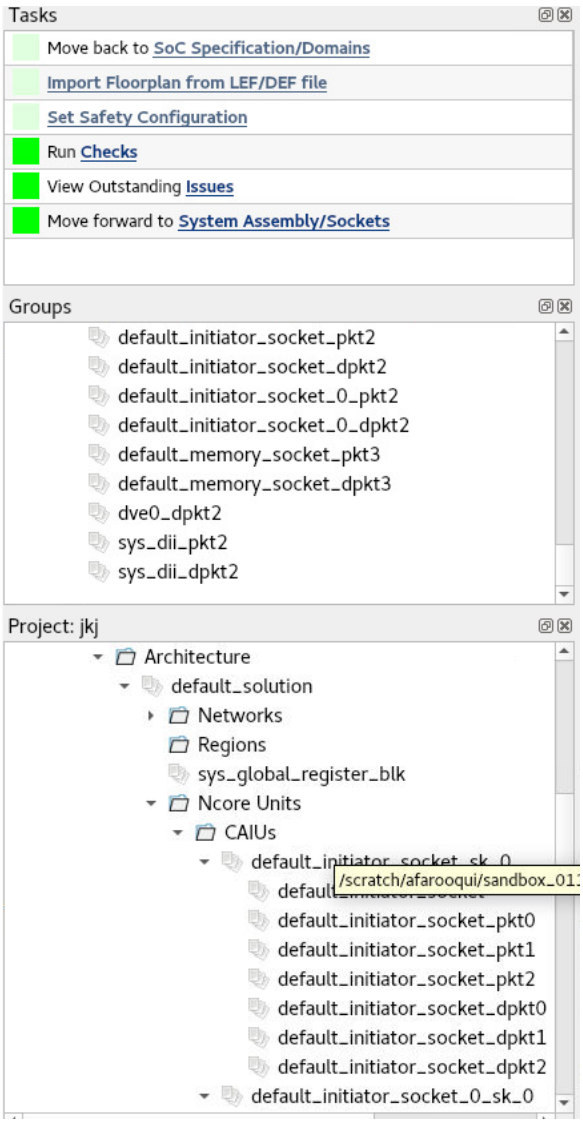


4. Select Transport solution (default_solution) object from the Project Tree, and select (activate) the **Native interface protection** parameter.

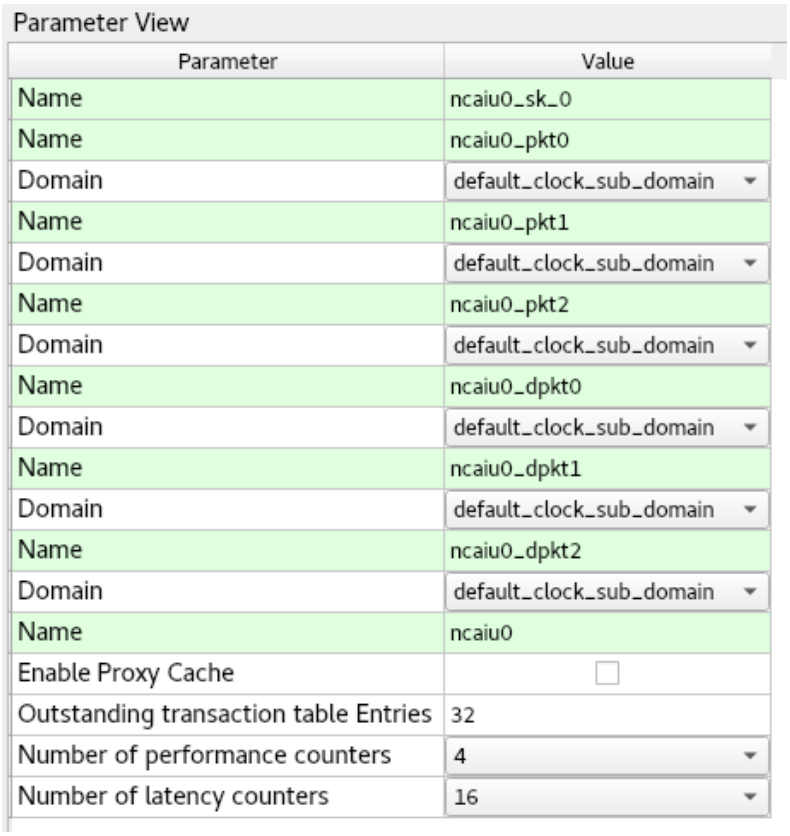


5. **Placeholder** objects will be created under the relevant interface units.

6. Select *safetyconfig* object from Project Tree and set *Current ASIL Level* to **NO_ASIL.**



7. Maestro will move back to the **Safety** stage. Placeholder objects will be removed from the interface units.


Parameter View

Parameter	Value
Name	ncaiu0_sk_0
Name	ncaiu0_pkt0
Domain	default_clock_sub_domain
Name	ncaiu0_pkt1
Domain	default_clock_sub_domain
Name	ncaiu0_pkt2
Domain	default_clock_sub_domain
Name	ncaiu0_dpkt0
Domain	default_clock_sub_domain
Name	ncaiu0_dpkt1
Domain	default_clock_sub_domain
Name	ncaiu0_dpkt2
Domain	default_clock_sub_domain
Name	ncaiu0
Enable Proxy Cache	
Outstanding transaction table Entries	32
Number of performance counters	4
Number of latency counters	16

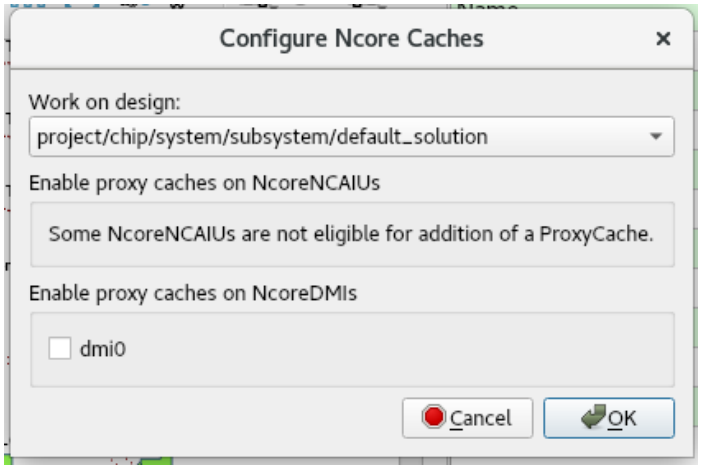
8. Advance Maestro to **Structural Design/Topology Structure** using the Task list.
9. Maestro will be back in the Structural design stage for the next step.

10. Use the *Project Tree* to select Interface unit **ncaiu0_sk_0.**

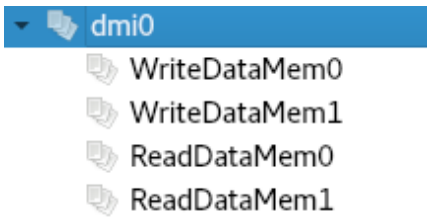
Parameter View

Parameter	Value
Name	ncaiu0_sk_0
Name	ncaiu0_pkt0
Domain	default_clock_sub_domain
Name	ncaiu0_pkt1
Domain	default_clock_sub_domain
Name	ncaiu0_pkt2
Domain	default_clock_sub_domain
Name	ncaiu0_dpkt0
Domain	default_clock_sub_domain
Name	ncaiu0_dpkt1
Domain	default_clock_sub_domain
Name	ncaiu0_dpkt2
Domain	default_clock_sub_domain
Name	ncaiu0
Enable Proxy Cache	☑
Outstanding transaction table Entries	32
Assigned SnoopFilter	<not set>
Number of performance counters	4
Tag Banks	1
Number of latency counters	16
Data Banks	1
Cache Replacement Policy	RANDOM

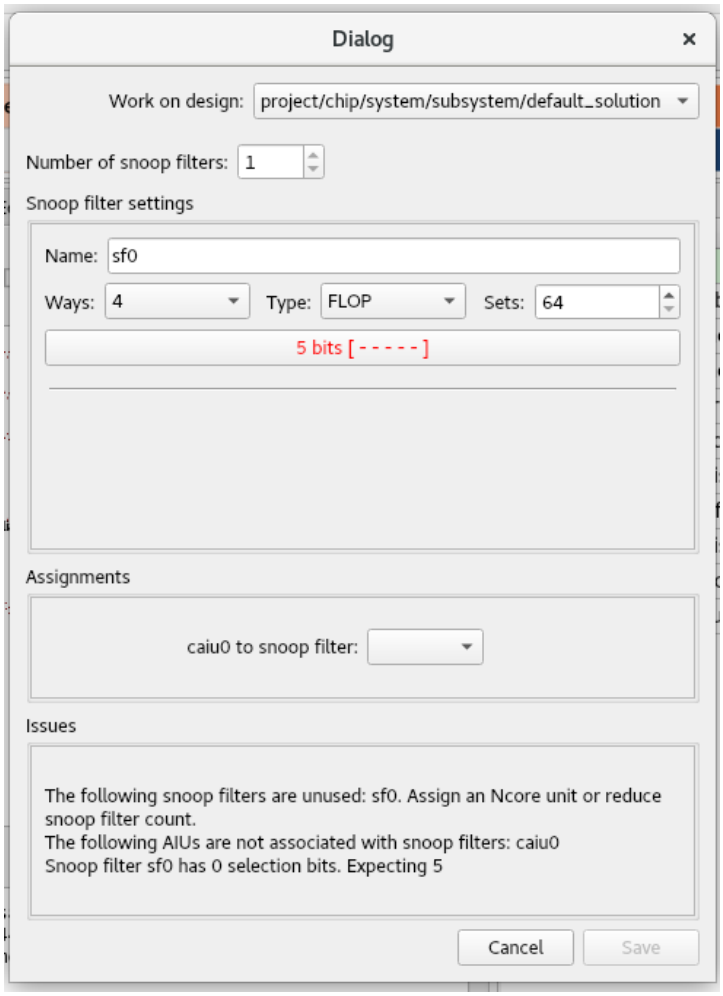
11. Use the *Parameter View* to **Enable Proxy Cache** for **ncaiu0_sk_0.**
12. Use the *Tasks list* to **Configure the Ncore Caches**.



13. Use the *Configure Ncore Caches dialog* to **enable memory caches** for **dmio.**

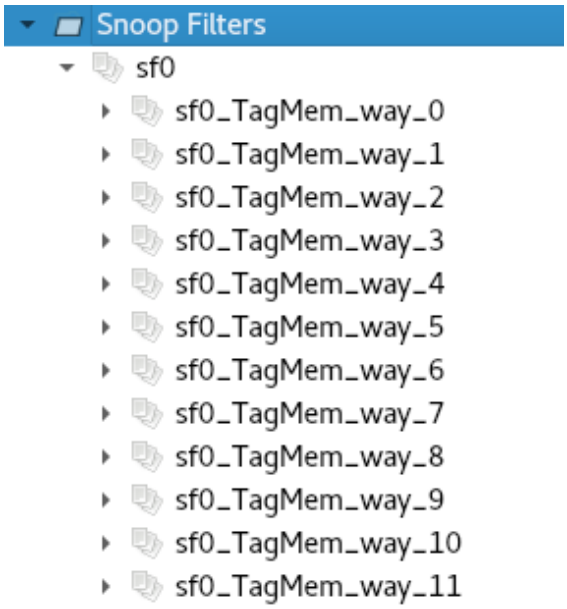


14. Use the Tasks list to Configure the Snoop Filters.

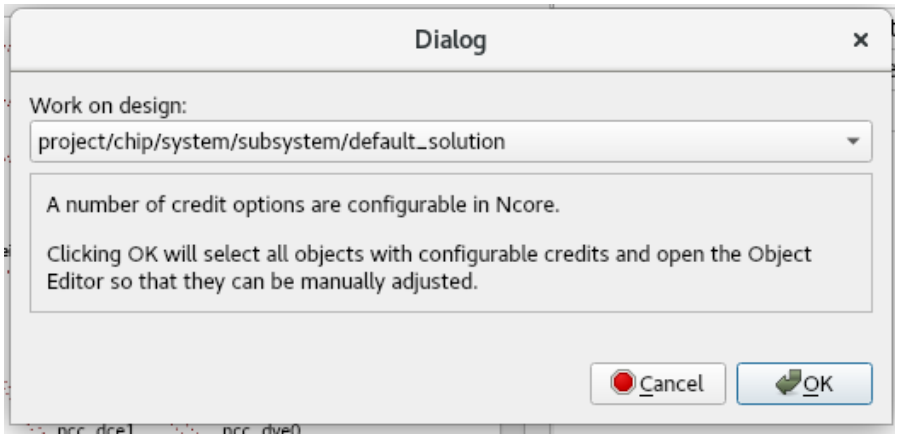


15. Use the *Configure Snoop Filters dialog* to configure 1 snoop filter:
    a. Ways: 12
    b. Type: SRAM
    c. Sets: 2048
    d. Primary bits set to 7-16 : 7 8 9 10 11 12 13 14 15 16

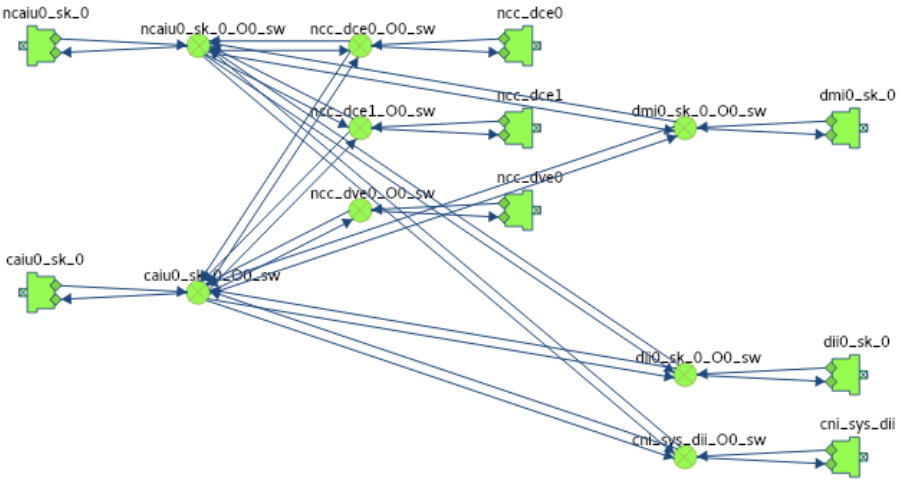
16. Assign **caiuo** and **ncaiuo** to snoop filter **sfo.**



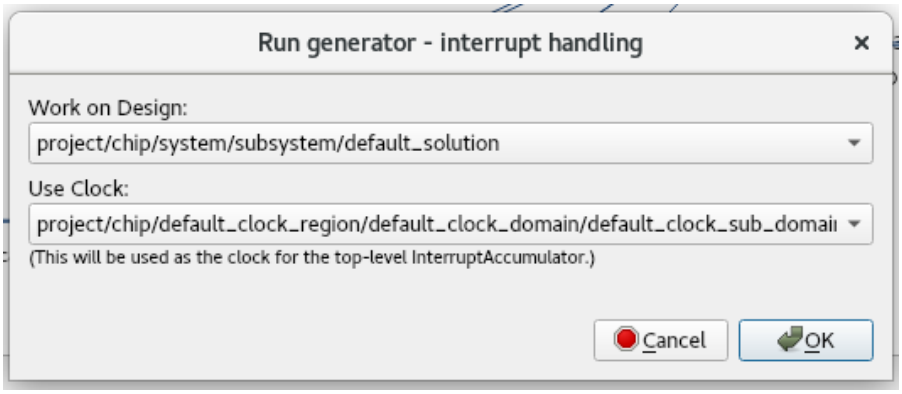
17. Use the *Tasks list* to **Configure the Ncore credits**.



18. Use the *Tasks list* to **Create a topology for Network "ndn1" by Preparing it for Manual Editing.**



19. Repeat the previous step for Networks "**ndn2**" and "**dn**".
20. The topologies are now created.
21. Use the *Tasks list* to **Insert Interrupt Handling**.




```
Info: [REQ-50] The 'csr' automation completed successfully
Info: [TSK-1030] Task insert_configuration_network successfully performed
```

22. Use the *Tasks list* to **Insert Configuration Network**.
23. Use the *Tasks list* to **Insert Power Management**.
24. Click **OK** to run the generator.
25. Use the *Tasks list* to navigate to **Mapping/Mapper**.

9.5 Phase 4 - Mapping

Goals for this phase: Mapper (refer to "Mapping (Phase 4)" on page 61.

1. Mapping is done automatically.

2. Use the *Tasks list* to navigate to **Export/Export**.

9.6 Phase 5 - Refinement

This phase is skipped in this design (refer to “Architectural Refinement (Phase 5)” on page 63).

9.7 Phase 6 - Export

Goals for this phase: Export RTL (refer to “Export Design (Phase 6)” on page 67).

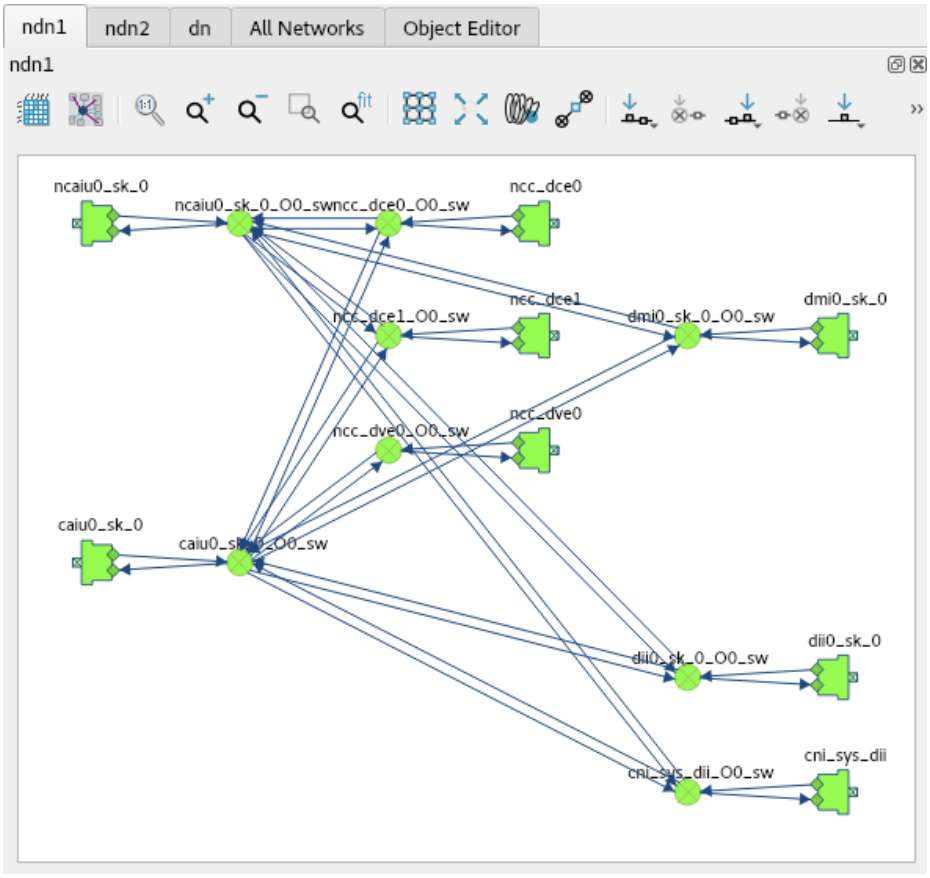
1. Click Generate RTL.
2. Make selections for exporting the design (data set, output path).
3. When finished, click Export.
4. In the Console, you should see a message indicating a successful export.
5. Save your design. The next section contains a topology exercise that can be used with this design.

9.8 Topology Editing Exercise

This exercise uses the design in the previous section to demonstrate how to use the topology editor and other tools. If you completed and saved the design, you can open the design and move to step 2.

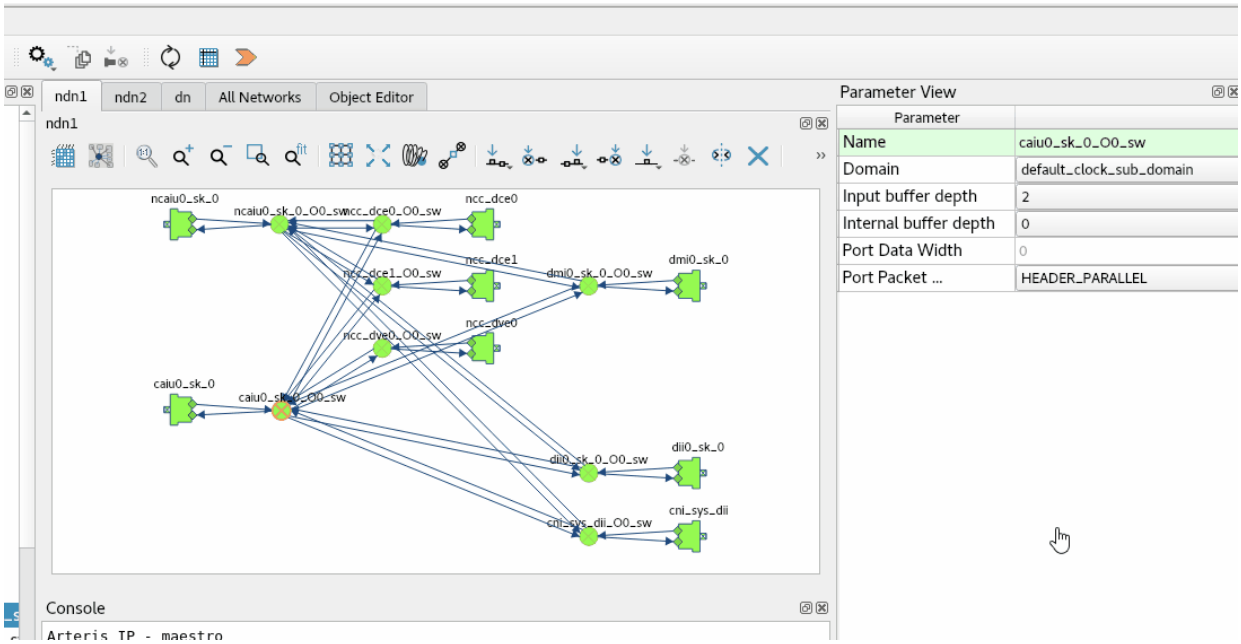
1. Run the demonstration design starting with “Phase 1 - SoC (Chip) Specification”.
2. Use the *Flow Navigator* to navigate to **Structural Design/Topology Structure**.

3. Use the *Dock Widgets* to display **ndn1.**



4. *Select* a switch (*caiuo_sk_o_Oo_sw*) so that it is orange highlighted and use the action **Delete selected items**.

Actions may either be triggered using the right-click menu or by clicking actions in the tool bar.



5. Use the **Undo** action.

   The switch is returned, and previously disconnected routes are reestablished

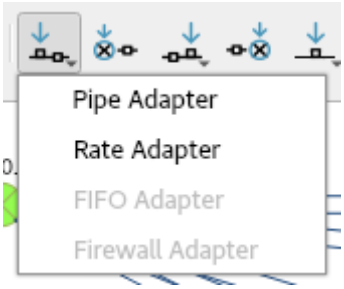
6. Use the **Redo** action.

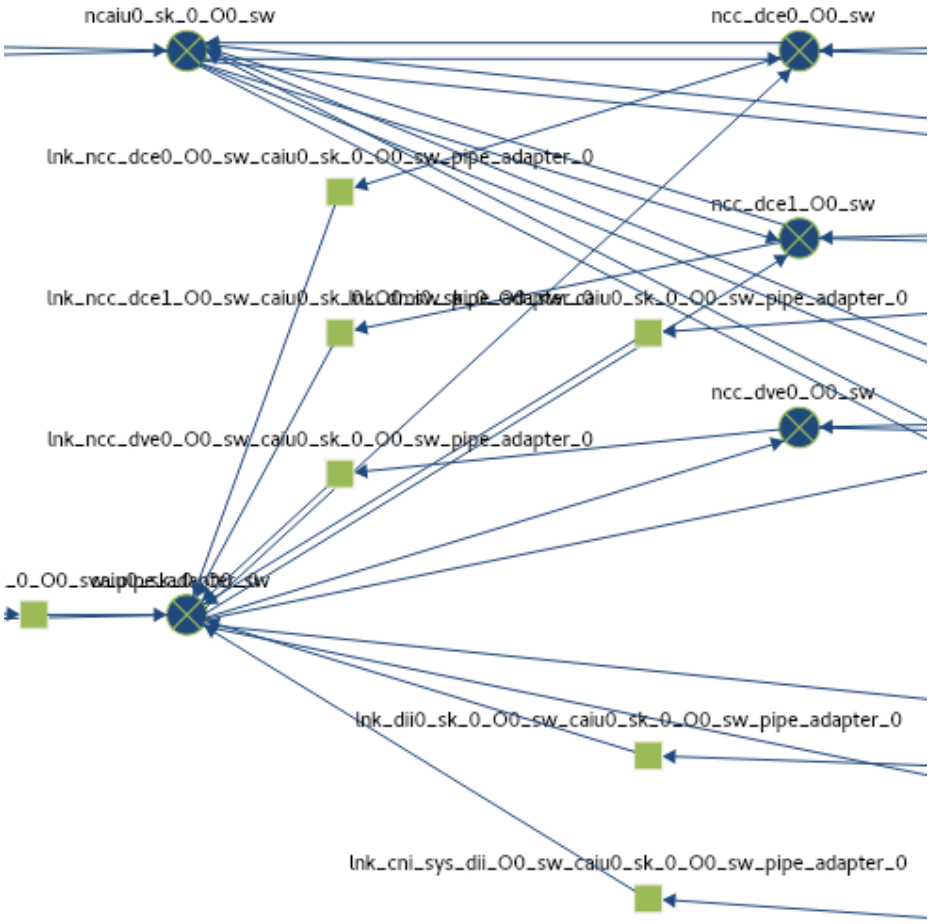
   The switch is deleted from the network and disconnected routes are displayed in red.

7. Use *Tasks list* to **Create a topology for Network "ndn1" by: Preparing it for Manual Editing**.

   The switch is returned, and previously disconnected routes are reestablished.

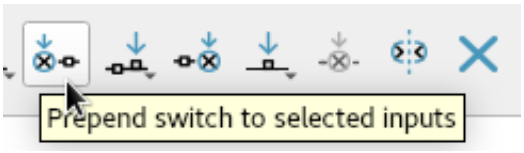
8. *Select* a switch (*caiu0_sk_0_O0_sw*) and use the action **Prepend Adapter to Selected Inputs->Pipe Adapter**.



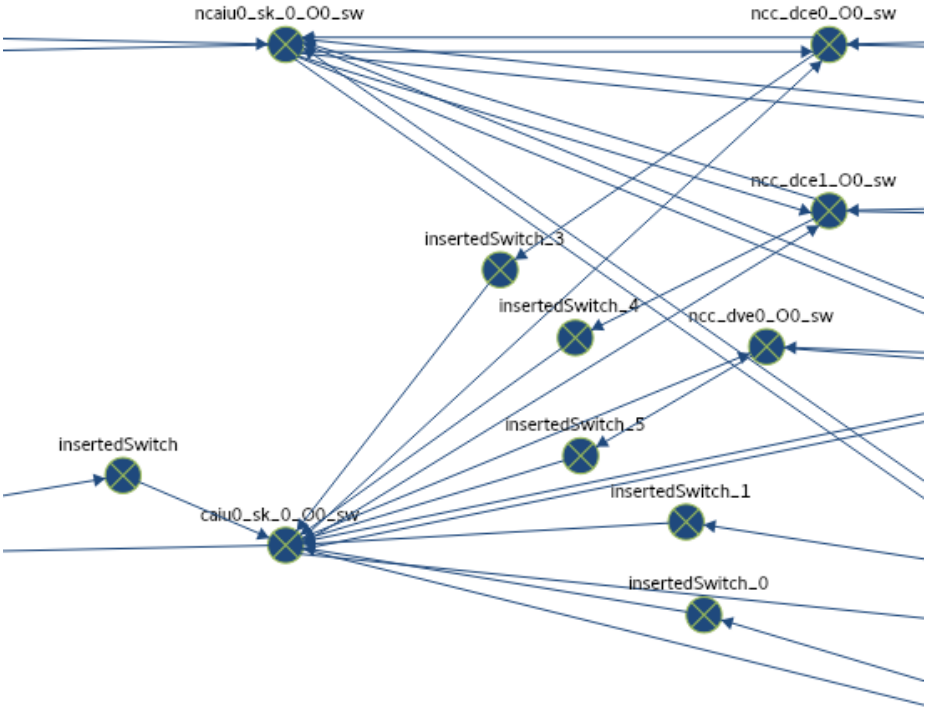


9. Use the **Undo** action.

   The Pipe Adapters are removed.



10. Select a switch (*caiuo_sk_o_Oo_sw*) and use the action **Prepend switch to selected inputs.**



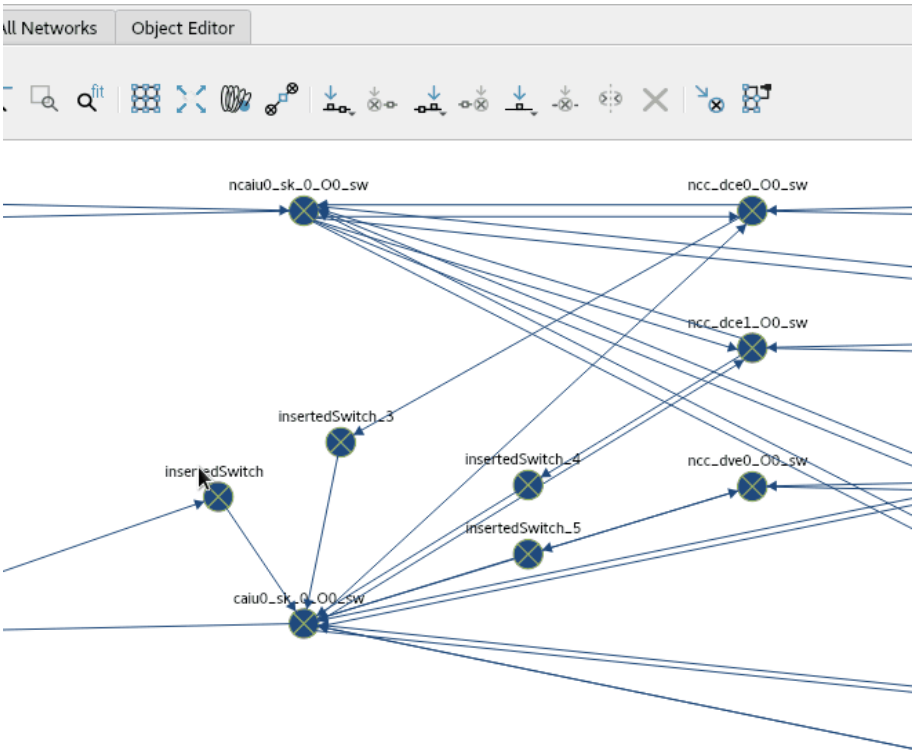
11. Use the **Undo** action.

    The Switches are removed

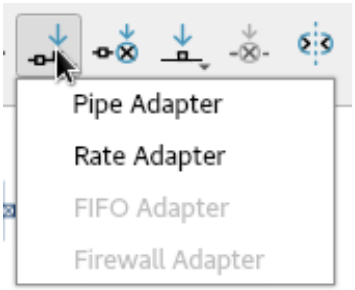
12. Use the **Redo** action.

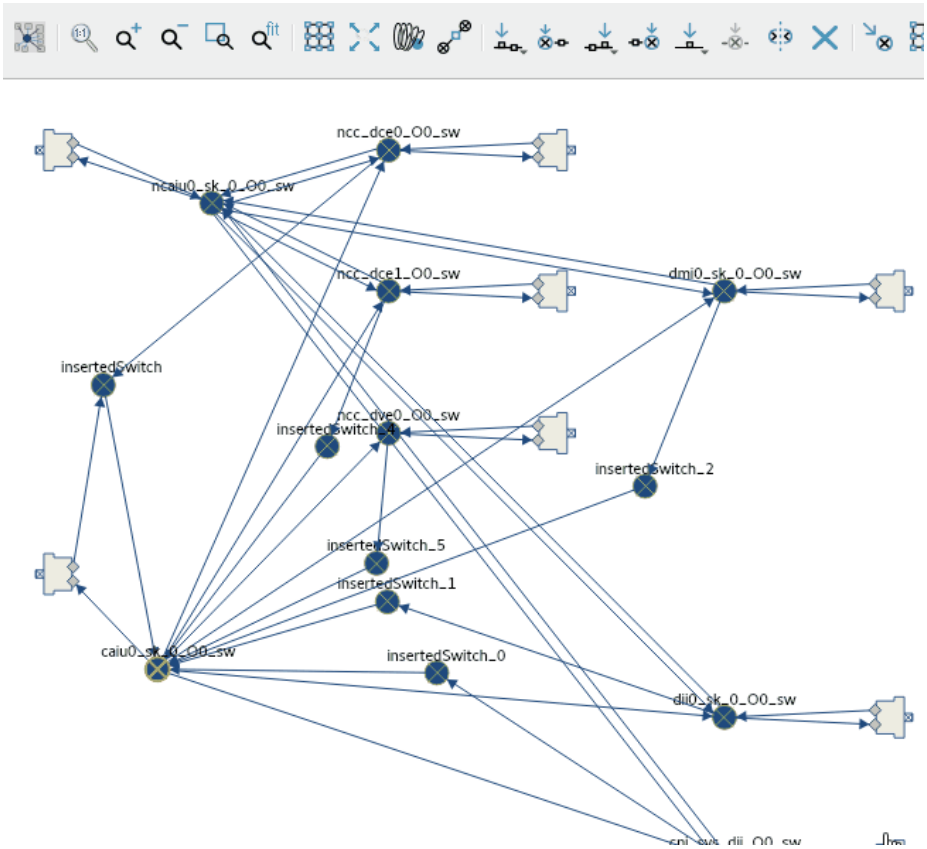
    The Switches are prepended to the switch's input.

13. Select an *inserted switch* and *another switch* and use the **Merge switch** action.



14. Select a switch (*caiuo_pktoZo_sw*) and use the action **Append Adapter to Selected Outputs->Pipe Adapter**.

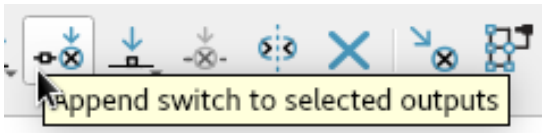


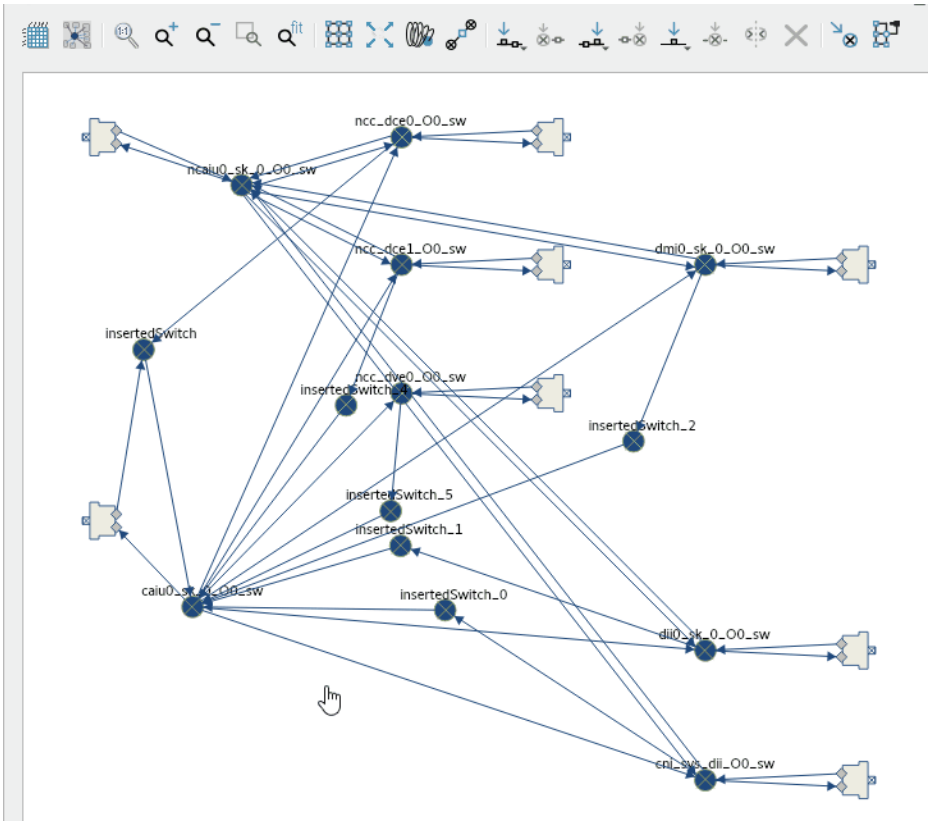


15. Use the **Undo** action.

    The adapters are removed

16. Select a switch (*caiu0_sk_0_O0_sw*) and use the action **Append switch to selected outputs**.

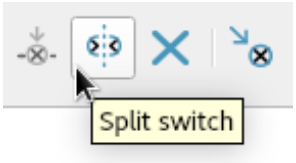


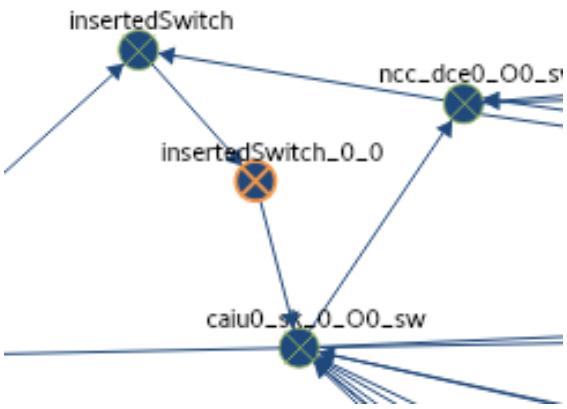


17. Use the **Undo** action.

    The switches are removed.

18. Select a *switch* and use the action **Split switch**.

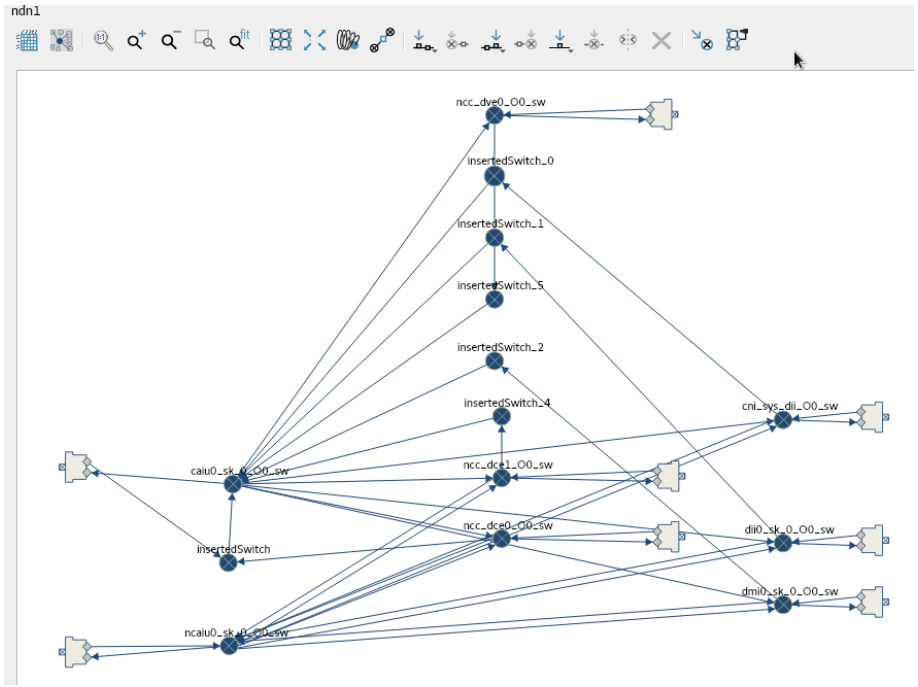




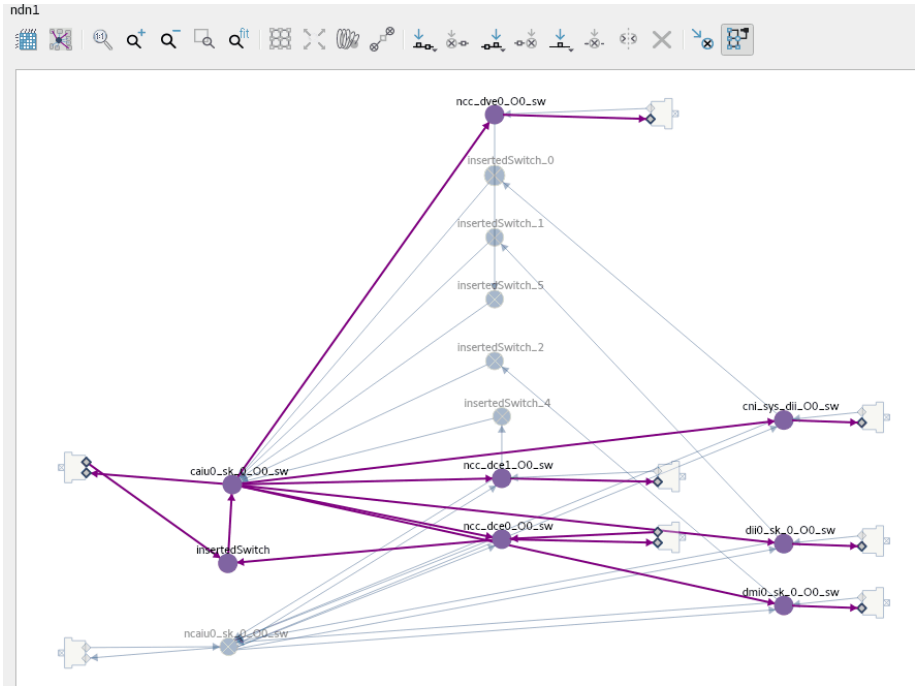
19. Use the **Undo** action.

    The switch is removed.

20. Use the action **Move (selected) Routes** and select a switch (*insertedSwitch*).



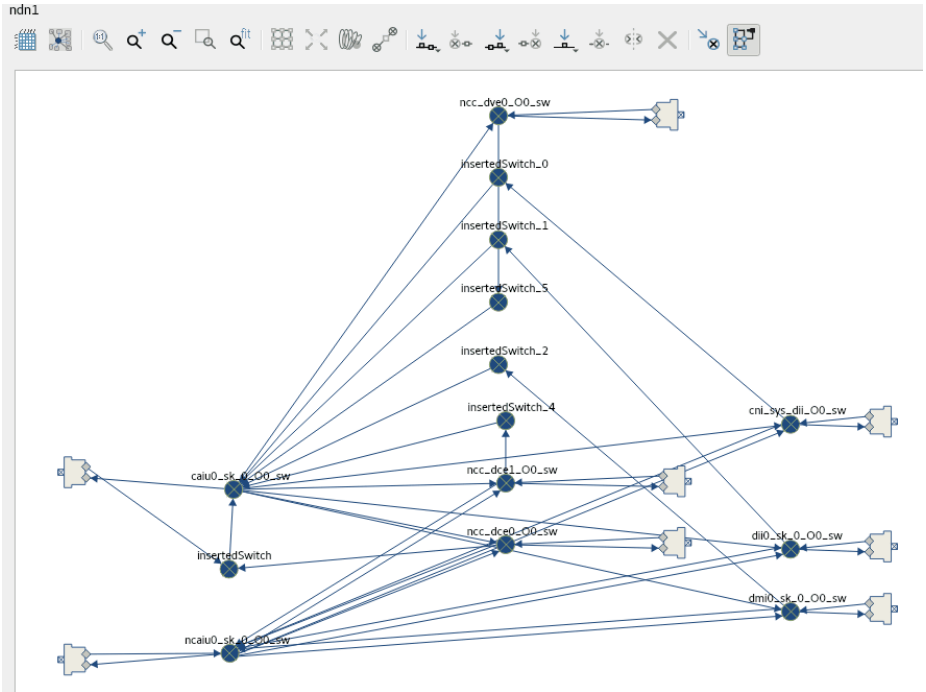
21. Drag the switch to an available switch (*insertedSwitch_0*) and release.



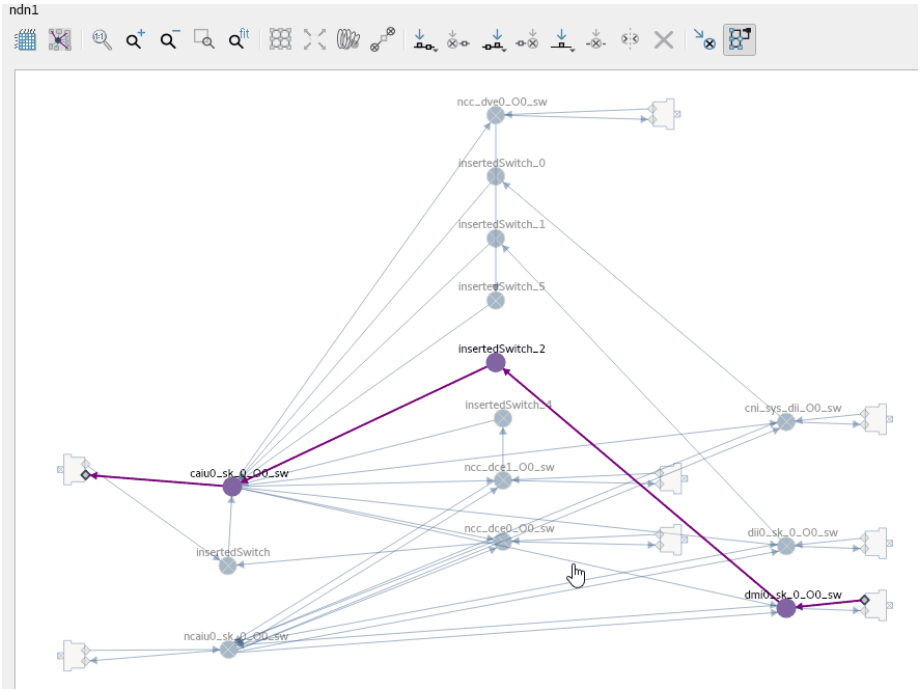
22. Use the **Undo** action.

The switch is returned, and the system remains in Move selected routes mode.

23. *Refine route selection* by **control-clicking** on a switch in the purple selected network (*insertedSwitch_2*).

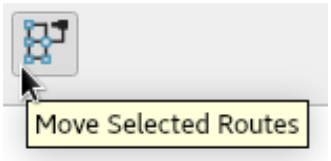


24. *Move the selected route* from the second selected switch (*insertedSwitch_2*) to another switch by dragging the second selected switch (*insertedSwitch_2*) to another switch (*insertedSwitch_0*).

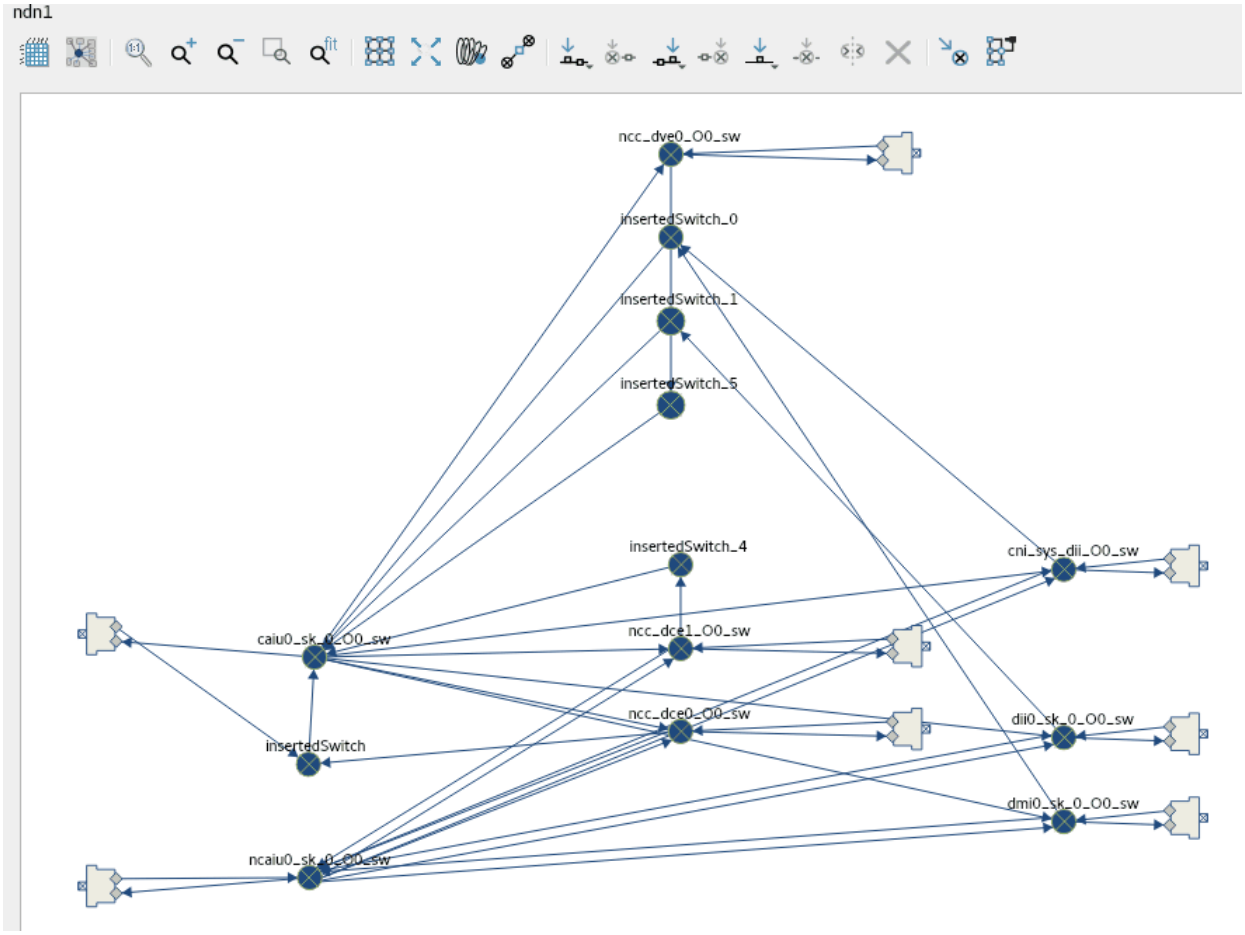


25. Click on **Move Selected Routes** button in the topology editor tool bar.

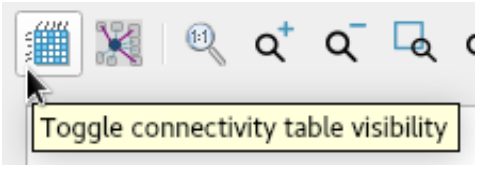
The application exits move routes mode.

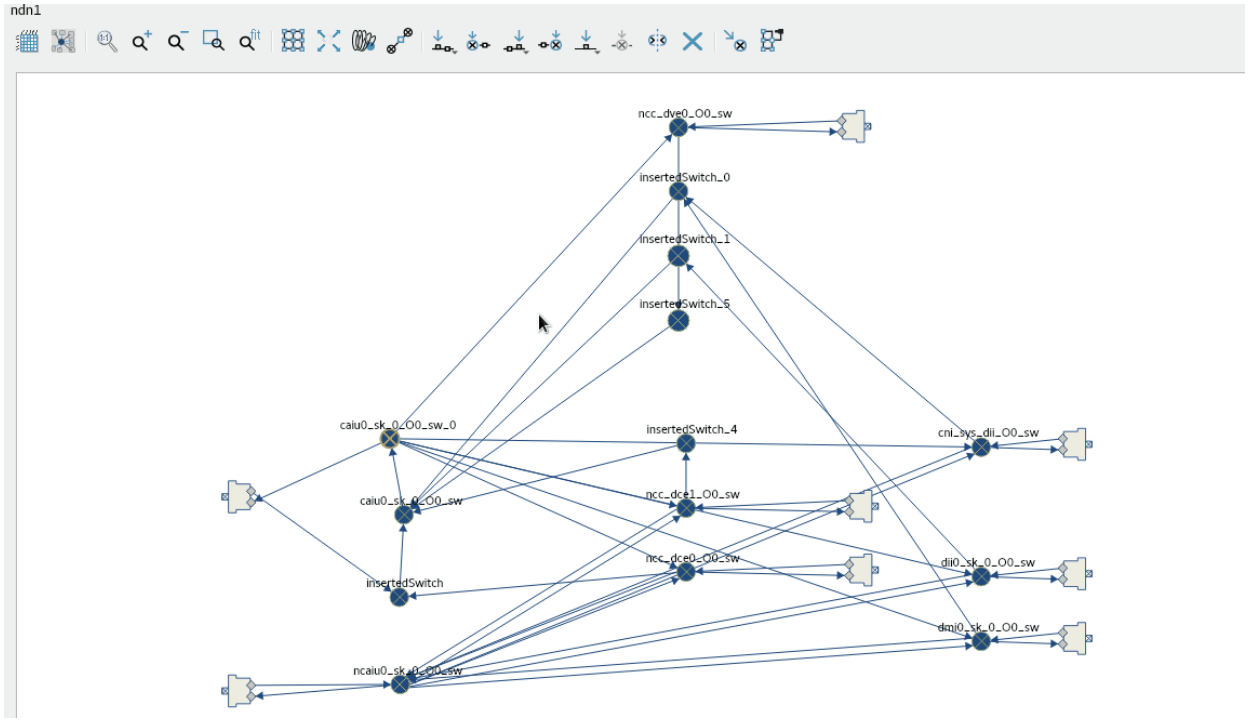


26. *Split a first hop switch* (*caiuo_pktoZo_sw*) by selecting and using the split switch action.

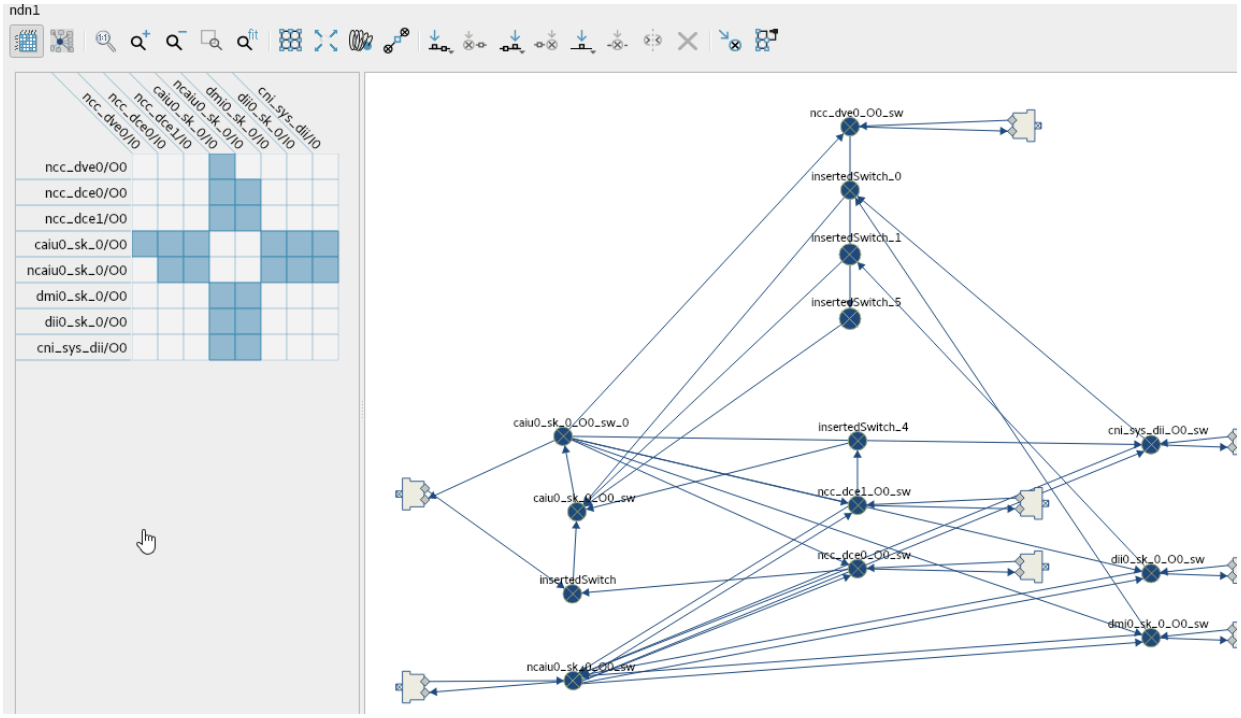


27. Toggle the connectivity table by clicking the **Toggle connectivity table visibility** button.

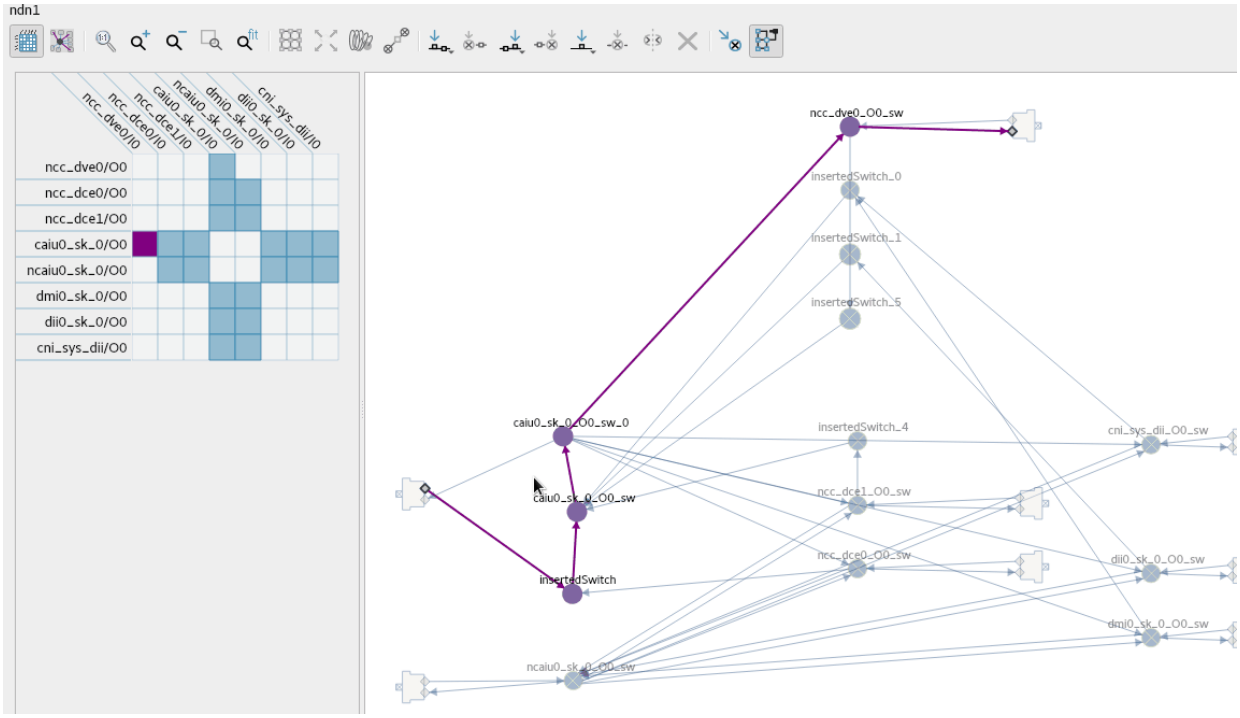




28. Use the *connectivity table* to select a single route (*caiu0_sk_0/Oo x ncc_dce0/I0*).



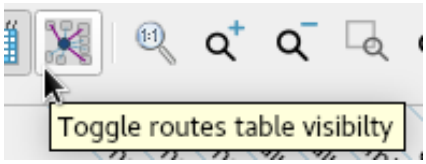
29. *Move the selected route* to a new switch by dragging from a non-first-hop switch (*caiuo_pktoOo_sw_o*) to another switch (*insertedSwitch_o*).

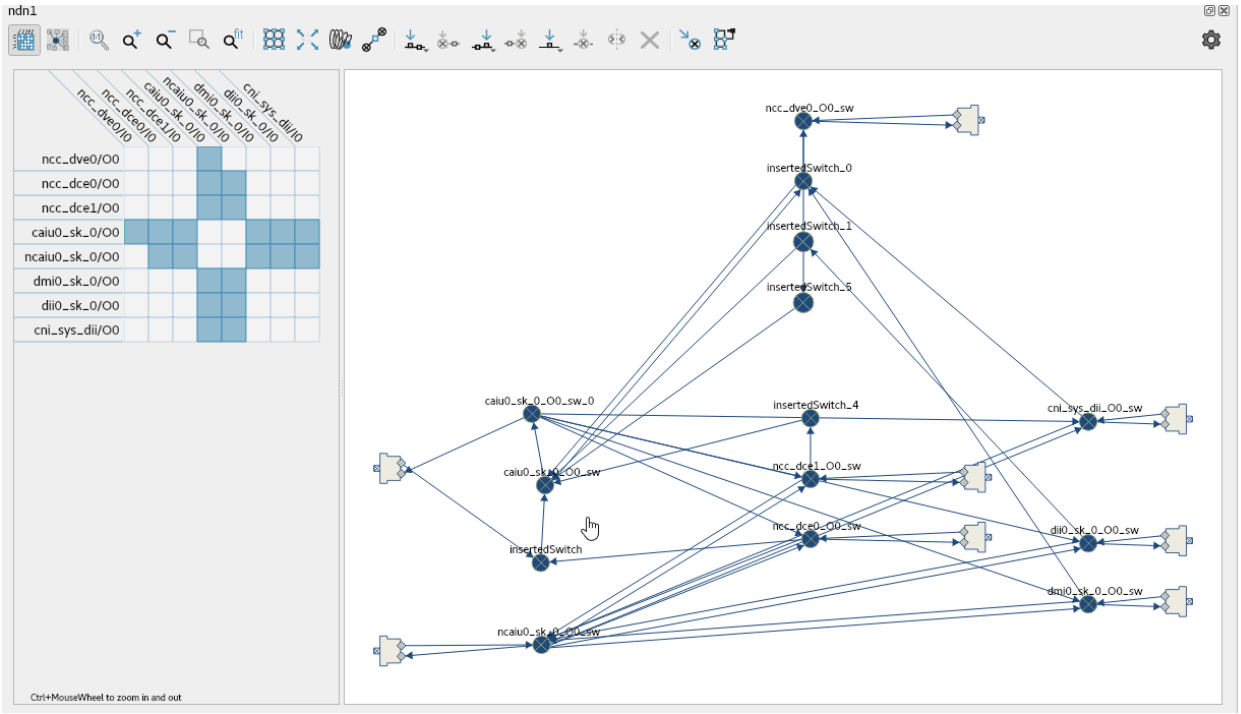


30. Click on **Move Selected Routes** button in the topology editor tool bar.

    The application exits move routes mode.

31. *Toggle the routing table* by clicking the **Toggle routes table visibility** button.





32. End of exercise.

10

Maestro Parameters

This chapter describes the Ncore parameters used by Maestro.

Object	DocID	Title	Tcl	Type	Description	Allowed Vals	Default
		Common Socket Parameters					
	193	Direction	direction	Str	Port Direction		IN
	383	Domain	domain	Hier Str	Clock subdomain which is associated with the socket		
	385	Function	socketFunction	Str	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. - MULTI_INITIATOR for multi-ported Initiators. - MEMORY for targets connected to memories, only for MASTERs. - PERIPHERAL for targets connected to peripherals, only for MASTERs.		INITIATOR
	384	Protocol	protocolType	Str	Protocols supported by ARM		<not set>
	10710	CSR Access	fnCsrAccess	Bool	Whether this Socket will be allowed access to the CSR network, if one is inserted. Only applicable to INITIATOR Sockets.		TRUE

Object	DocID	Title	Tcl	Type	Description	Allowed Vals	Default
	3456	Number of Sockets	cardinality	Int	Number of identical sockets		1
	1400	Supports Wrapping Bursts	wrapSupported	Bool			FALSE
	1401	Supports Narrow Bursts	narrowSupported	Bool			FALSE
	1402	Supports Fixed Bursts	fixedSupported	Bool			FALSE
	1403	Max Burst Length	maxBurstLength	Int			1
	1404	Supports Reads	readSupported	Bool			TRUE
	1405	Supports Writes	writeSupported	Bool			TRUE
	1406	Supports Read Interleaving	readInterleaveSupported	Bool			FALSE
	1777	Disable Read Interleave	fnDisableRdInterleave	Bool	When set disables read data interleaving across different AXI IDs.		FALSE
	116	Sockets This Depends On	socketsThisDependentOn	Hier Str	A list of Sockets that this Socket's transactions may depend on. For example, if a single RTL unit has both an INITIATOR Socket 'I' and a MEMORY Socket 'M', M may depend on I because a limited-depth internal buffer needs to have sufficient free space before a transaction on M can be accepted, and this space has to be freed by an I transaction being performed. In this case, I should be added to the dependency list of M to allow Maestro to properly detect possible system-level deadlock conditions.		
	117	Sockets Dependent On This	socketsDependentOnThis	Hier Str	Based on Socket dependencies specified under 'Sockets This Depends On', this parameter presents the inverse dependency list, namely, the list of Sockets that have a possible communication dependency on this Socket.		
	118	Initiator Supports Early Response	earlyWriteResponseSupported	Bool			FALSE
		Socket with AXI4					
	227	Width of Ar Id	wArId	Int	Specify the width of AXI interface ArId bits.	Min: 1, Max: 32	6
	228	Width of Aw Id	wAwId	Int	Specify the width of AXI interface AwId bits.	Min: 1, Max: 32	6

Object	DocID	Title	Tcl	Type	Description	Allowed Vals	Default
	229	AXI address bus width	wAddr	Int	Specify the width of AXI interface address bits.	Min: 12, Max: 64	32
	231	AXI data bus width	wData	Int	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		64
	966	Width of Aw user	wAwUser	Int	Specify the width of AXI interface Aw user bits.	Min: 0 Max: 32	0
	967	Width of W user	wWUser	Int	Specify the width of AXI interface w user bits.	Min: 0 Max: 1280	0
	968	Width of B user	wBUser	Int	Specify the width of AXI interface b user bits.	Min: 0, Max: 64	0
	969	Width of Ar user	wArUser	Int	Specify the width of AXI interface Ar user bits.	Min: 0, Max: 32	0
	970	Width of R user	wRUser	Int	Specify the width of AXI interface R user bits.	Min: 0, Max: 1280	0
	971	Width of size	wSize	Int		Min: 3, Max: 4	3
	972	Width of region	wRegion	Int		Min: 0, Max: 4	0
		Socket with ACE					
	951	width of address	wAddr	Int	Width of the address bits.		32
		Socket with ACE-Lite					
	73	Enable DVM	enableDVM	Bool	Add AC channel and enable DVM support.		FALSE
		Socket with ACE-Lite-E					
	964	Enable DVM	enableDVM	Bool	Add AC channel and enable DVM support.		FALSE
		Socket with CHI-A					
	898	Width of source Id	SrcID	Int	Width of the Source ID.	Min: 7, Max: 7	7
	899	Width of target Id	TgtID	Int	Width of the Target ID.	Min: 7, Max: 7	7
	900	CHI address bus width	wAddr	Int	Width of the CHI address bus.	Min: 44, Max: 44	44
	105	Width of request RSVDC	REQ_RSVDC	Int	Width of the request RSVDC/ user bits.		0
	107	CHI data bus width	wData	Int	Width of data on the CHI interface.		128
		Socket with CHI-B					

Object	DocID	Title	Tcl	Type	Description	Allowed Vals	Default
	109	Width of Node Id	NodeID_Width	Int	Width of the Node ID of the CHI Interface.	Min: 7, Max: 11	7
	961	Width of source Id	SrcID	Int	Width of the Source ID	Min: 7, Max: 11	7
	962	Width of target Id	TgtID	Int	Width of the Target ID	Min: 7, Max: 11	7
	110	CHI address bus width	wAddr	Int	Width of the address on CHI interface.	Min: 44, Max: 52	48
	113	Width of request RSVDC	REQ_RSVDC	Int	Width of the request RSVDC/ user bits.		0
	115	CHI data bus width	wData	Int	Width of data on the CHI interface.		128
	963	Enable poison bits	enPoison	Bool	Enable poison port support on CHI interface		FALSE
		Memory Interface					
	1046	Memory RTL name	rtlPrefixStr	Str	Specify the name of the embedded memory module to instantiate. The system designer must provide the module to be instantiated with this name.		
	1047	Memory Type	MemType	Str	Specify the choice of memory implementation, between a Register (flip-flop) implementation, or an embedded memory IP core instantiation. Ncore uses Synopsys embedded memory port names and timing.		NONE
	887	Name of the signal	Name	Str	Name of the soc signal that is to be routed and connected to the memory instance.		
	888	Direction of the signal	Direction	Str	Direction of the signal.		
	889	Width of the signal	Width	Int	Width of the signal.	Min: 1, Max: 128	
		Clock					
	21	Power Region	powerRegion	Hier Str	Power Region this object is inside of.		
	22	Frequency	frequency	Int	Frequency of the input clock signal of the Clock Region.	Min: 1, Max: 1000000000	500000
	24	Unit-Level Clock Gating	unitClockGating	Bool	Whether it is desired that units receiving a clock signal from this Clock Region subdomains should be subject to unit-level clock gating, when possible, for power savings. Disabling this optimization might help timing.		FALSE
	19	Power Domain	powerDomain	Hier Str	Power Domain this object is inside of.		

Object	DocID	Title	Tcl	Type	Description	Allowed Vals	Default
	20	Gating	gating	Str	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. Select the option 'always_on' if no gating should be applied, or 'external' if logic external to the interconnect can gate this clock.		always_on
		Power					
	61	Voltage	voltage	Int	Voltage of the power supply of the Power Region.	Min: 1, Max: 25000	950
	60	Gating	gating	Str	Whether the voltage supply specified in the parent Power Region should be subject to disconnection when sent to a portion of the chip, that shall 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 shall also be modeled as its own Power Domain.		always_on
		Initiator Group					
	1966	Initiator Sockets	initiators	Hier Str	Sockets part of the InitiatorGroup		
	1967	Interleaving Bits (Left is LSB)	interleavingBits	Int	Bits on which the initiators in the InitiatorGroup are interleaved (Left most bit is LSB)		
		Non Coherent Network Interface					
	348	Input Packet Port	inPort	Hier Str	Input packet port		
	349	Output Packet Port	outPort	Hier Str	Output Packet Port		
		Packetizer					
	778	ATP-pipe Depth	pipeLevelAtp	Int	Depth of ATP pipes inside SMI	Min: 0, Max: 2	2
	779	SMI-HDR-pipe Depth	pipeLevelSmiHdr	Int	Depth of SMI HDR pipe inside SMI	Min: 0, Max: 2	2
	780	SMI-Data-pipe Depth	pipeLevelSmiDP	Int	Depth of SMI DP pipe inside SMI	Min: 0, Max: 2	2
		Ncore Settings					
	1299	DCE Count	dceCount	Int	Number of NcoreDCEs (directories) in the design.		1

Object	DocID	Title	Tcl	Type	Description	Allowed Vals	Default
		Boot Info					
	1297	Target Socket	physicalChannel	Hier Str	Region maps to one target (or an aggregate of multiple identical targets for address striping).		
	279	Base Address	memoryBase	Int	The base address of the region.	Min: 0	
	280	Size	memorySize	Int	Size of the memory. Each region size is power of 2, starting address is aligned to its size.	Min: 0	
	1298	Target DynamicMemory Group	memoryGroup	Hier Str	This parameter defines the memoryGroup this bootregion can be accessed from.		
	1	Boot Region Home Unit ID	regionHui	Int	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	Boot Region Size	regionSize	Int	specifies the size of boot region. Minimum is 4KB and must be a power of two.		4096
	3	Boot Region Home Unit Type	regionHut	Str	Specify the home unit Type associated with the boot region. This can be either DMI (Memory) or DII (IO device).		DMI
		Clock Adapter					
	989	Output Pipe	pipeOut	Bool	Whether a pipe should be instantiated on the output of the component, to improve timing.		TRUE
	489	Output Clock Domain	outputDomain	Hier Str	Output Clock Domain		
		Configuration Region					
	1297	Target Socket	physicalChannel	Hier Str	Region maps to one target (or an aggregate of multiple identical targets for address striping).		
	279	Base Address	memoryBase	Int	The base address of the region.	Min: 0	
	280	Size	memorySize	Int	Size of the memory. Each region size is power of 2, starting address is aligned to its size	Min: 0	
		Flow					
	4	Initiator	source	Hier Str	Initiator of the flow		
	6	Target	destination	Hier Str	Target of the flow		
	506	Bandwidth	bandwidth	Int	Bandwidth requirements of the Flow	Min: 0, Max: 1073741824	1048576

Object	DocID	Title	Tcl	Type	Description	Allowed Vals	Default
		Generic Port					
	193	Direction	direction	Str	Port Direction		IN
	508	Port Width	wireWidth	Int	Port width for generic port	Min: 1, Max: 1024	1
		Group					
	2543	Physical Region	region	Hier Str	Physical Region associated with this group.		
	2544	Preferred CSR clock	preferredCSRDomain	Hier Str	Preferred CSR Clock Subdomain.		
	2545	Generate Hierarchy	generateHierarchy	Bool	Whether to generate a layer of logical hierarchy to the RTL.		
		Initiator Transform					
	143	Narrow Initiator	initiator	Hier Str	Narrow Initiator		
	144	Memory Offset for the Narrow Initiator	memoryOffset	Int	Memory Offset for the Narrow Initiator.	Min: 0	0
		Memory					
	96	Memory Type	memoryType	Str	This parameter is used to set the type of memory. This parameter is an enumerated type whose values are either FLOP or SRAM.		FLOP
		Memory Map					
	241	System Directory primary set select bits (Left is LSB)	dceInterleavingBits	Int	Address bits to be used as interleaving bits for DCE selection. (Left most bit is LSB.)	Min: 6, Max: 64	6
	16	Target Sockets	physicalChannels	Hier Str			
	262	Primary Interleaving Bit #1 (LSB)	primaryInterleavingBitOne	Int	When DMI interleaving is configured via a DynamicMemoryGroup, this is the first bit index used for interleaving. When 2-way interleaving is configured, primaryInterleavingBitOne must be specified and valid.	Min: 0, Max: 8192	0
	264	Primary Interleaving Bit #2	primaryInterleavingBitTwo	Int	When 4-way interleaving is configured, primaryInterleavingBitOne and primaryInterleavingBitTwo must be specified. They must be unique and valid.	Min: 0, Max: 8192	0

Object	DocID	Title	Tcl	Type	Description	Allowed Vals	Default
	266	Primary Interleaving Bit #3	primaryInterleavingBitThree	Int	When 8-way interleaving is configured, primaryInterleavingBitOne and primaryInterleavingBitTwo and primaryInterleavingBitThree must be specified. They must be unique and valid.	Min: 0, Max: 8192	0
	268	Primary Interleaving Bit #4	primaryInterleavingBitFour	Int	When 16-way interleaving is configured, primaryInterleavingBitOne, primaryInterleavingBitTwo, primaryInterleavingBitThree and primaryInterleavingBitFour must be specified. They must be unique and valid.	Min: 0, Max: 8192	0
		Memory Region					
	1297	Target Socket	physicalChannel	Hier Str	Region maps to one target (or an aggregate of multiple identical targets for address striping).		
	279	Base Address	memoryBase	Int	The base address of the region.	Min: 0	
	280	Size	memorySize	Int	Size of the memory. Each region size is power of 2, starting address is aligned to its size.	Min: 0	
		Coherent Agent Interface Unit (CAIU)					
	339	Domain	domain	Hier Str	Clock SubDomain associated with this unit.		
	519	Assigned SnoopFilter	snoopFilter	Hier Str	Specify the snoop filter associated with this AIU.		
	520	Credits for Native Protocol	nNativeCredits	Int	Specify the maximum number of CHI link credits this AIU should support.	Min: 2, Max: 15	2
	521	Outstanding transaction table Entries	nOttCtrlEntries	Int	Specify the maximum number of outstanding native transactions this AIU should support.	Min: 8, Max: 256	8
	525	Stashing Snoop Credits	nStshSnpCredits	Int	Specify the maximum number of outstanding stash snoops this AIU should support. These are stash snoops issued on the CHI interface.	Min: 1, Max: 8	2
	162	Number of Processors	nProcessors	Int	Specify the number of logical processors connected to this interface.		1
	163	Number of performance counters	nPerfCounters	Int	Number of performance counters per Ncore unit.		4
	121	Number of latency counters	nLatencyCounters	Int	Number of latency counters in the Ncore unit.		16
		Distributed Coherence Enforcement Units (DCE)					

Object	DocID	Title	Tcl	Type	Description	Allowed Vals	Default
	339	Domain	domain	Hier Str	Clock SubDomain associated with this unit.		
	527	Number of DCE write request buffer credits	nDceRbCredits	Int	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.	Min: 2, Max: 32	2
	528	Number of snoop credits	nAiuSnpCredits	Int	Specify the maximum number of snoop request credits per AIU.	Min: 2, Max: 16	2
	530	Number of active coherent transactions	nAttCtrlEntries	Int	Specify the maximum number of active coherent transactions tracked by each DCE.	Min: 4, Max: 64	4
	11	Total depth of the Skid Buffer	nCMDSkidBufSize	Int	Total depth of skid buffer for commands to DCE/DII and the non-coherent port of DMI. The skid buffer is used to stage transaction requests from initiator agents. The number of required entries may be determined by traffic requirements and analysis using performance modeling. This value sets the total budget of protocol credits available for distribution.	Min: 4, Max: 320	16
	12	Depth of skid buffer visible to arbitration	nCMDSkidBufArb	Int	Depth of skid buffer visible to arbitration. This value determines the size of the arbitration window within which arriving requests are selected based on QoS, priority and arrival time. It is recommended to start with a reasonably value for performance analysis -the area of a skid buffer grows with the square of this number and larger options will also significantly impact timing.		16
	191	Number of performance counters	nPerfCounters	Int	Number of performance counters per Ncore unit.		4
	174	Number of Tagged exclusive monitors	nTaggedMonitors	Int	Specify the desired number of tagged exclusive monitor per DCE instance. Note that each DCE instance will always have a basic exclusive monitor.	Min: 0, Max: 8	0
		Distributed IO Interface (DII)					
	339	Domain	domain	Hier Str	Clock SubDomain associated with this unit.		
	531	Number of Write request buffer credits	nDiiRbCredits	Int	Specify the maximum number of non coherent write request buffer credits.	Min: 2, Max: 32	2

Object	DocID	Title	Tcl	Type	Description	Allowed Vals	Default
	532	Max outstanding read transactions	nRttCtrlEntries	Int	Specify number of outstanding read transactions on the AXI interface	Min: 4, Max: 32	4
	533	Max outstanding write transactions	nWttCtrlEntries	Int	Specify number of outstanding write transactions on the AXI interface	Min: 4, Max: 32	4
	8	Total depth of the Skid Buffer	nCMDSkidBufSize	Int	Total depth of skid buffer for commands to DCE/DII and the non-coherent port of DMI. The skid buffer is used to stage transaction requests from initiator agents. The number of required entries may be determined by traffic requirements and analysis using performance modeling. This value sets the total budget of protocol credits available for distribution.	Min: 4, Max: 320	16
	9	Depth of skid buffer visible to arbitration	nCMDSkidBufArb	Int	Depth of skid buffer visible to arbitration. This value determines the size of the arbitration window within which arriving requests are selected based on QoS, priority and arrival time. It is recommended to start with a reasonably value for performance analysis -the area of a skid buffer grows with the square of this number and larger options will also significantly impact timing		16
	52	Number of performance counters	nPerfCounters	Int	Number of performance counters per Ncore unit.		4
	121	Number of latency counters	nLatencyCounters	Int	Number of latency counters in the Ncore unit.		16
	1092	Size of the largest end point	nLargestEndPoint	Int	Specify the size of the largest endpoint device connected to the DII. The size is in KBs. This size issued to achieve endpoint ordering as defined by ARM CHI specification.	Min: 4, Max: 549755813888	4
	1095	Number of Address Translation Registers	nAddrTransRegisters	Int	Specifies the number of address translation registers that are avialble within the DII. Refer to Address Translation Capability in the Reference Guide.	Min: 0, Max: 4	0
		Distributed Memory Interface (DMI)					
	339	Domain	domain	Hier Str	Clock SubDomain associated with this unit		
	534	Number of DMI write buffers	nDmiRbCredits	Int	Specify the maximum number of non coherent write request buffer credits.	Min: 2, Max: 64	2

Object	DocID	Title	Tcl	Type	Description	Allowed Vals	Default
	535	Max outstanding memory read transactions (RTT Entries)	nRttCtrlEntries	Int	Specify number of outstanding read transactions on the AXI interface	Min: 4, Max: 128	4
	536	Max outstanding memory write transactions (WTT Entries)	nWttCtrlEntries	Int	Specify number of outstanding write transactions on the AXI interface.	Min: 4, Max: 64	4
	1534	Enable System Memory Cache	hasSysMemCache	Bool	Whether a System Memory Cache should be instantiated in the DMI.		FALSE
	1535	Tag Banks	nTagBanks	Int	Number of tag banks in the System Memory Cache.		1
	1536	Data Banks	nDataBanks	Int	Number of data banks in the System Memory Cache.		1
	1540	Cache Replacement Policy	cacheReplPolicy	Str	Cache Replacement Policy		RANDOM
	17	Total depth of the Skid Buffer	nCMDSkidBufSize	Int	Total depth of skid buffer for commands to DCE/DII and the non-coherent port of DMI. The skid buffer is used to stage transaction requests from initiator agents. The number of required entries may be determined by traffic requirements and analysis using performance modeling. This value sets the total budget of protocol credits available for distribution.	Min: 4, Max: 320	16
	23	Depth of skid buffer visible to arbitration	nCMDSkidBufArb	Int	Depth of skid buffer visible to arbitration. This value determines the size of the arbitration window within which arriving requests are selected based on QoS, priority and arrival time. It is recommended to start with a reasonably value for performance analysis -the area of a skid buffer grows with the square of this number and larger options will also significantly impact timing		16

Object	DocID	Title	Tcl	Type	Description	Allowed Vals	Default
	25	Total depth of skid buffer for coherent DMI transactions	nMrdSkidBufSize	Int	Total depth of skid buffer for coherent DMI transactions - arriving from DCE. The skid buffer is used to stage transaction requests from initiator agents. The number of required entries may be determined by traffic requirements and analysis using performance modeling. This value sets the total budget of protocol credits available for distribution.	Min: 4, Max: 320	16
	26	Depth of skid buffer visible to arbitration for coherent DMI transactions	nMrdSkidBufArb	Int	Depth of skid buffer visible to arbitration. This value determines the size of the arbitration window within which arriving requests are selected based on QoS, priority and arrival time. It is recommended to start with a reasonably value for performance analysis - the area of a skid buffer grows with the square of this number and larger options will also significantly impact timing		16
	120	Number of performance counters	nPerfCounters	Int	Number of performance counters per Ncore unit.		4
	121	Number of latency counters	nLatencyCounters	Int	Number of latency counters in the Ncore unit.		16
	185	Enable Atomic Engine	useAtomic	Bool	This option adds an atomic engien in DMI. It must be enabled when an atomic capable master is present in the system and requires atleast a 4KB SMC.		FALSE
	997	Dmi QoS threshold value.	DmiQoSThVal	Int	DMI QoS threshold value. Traffic with QoS equal to or above this value are considered as high priority hard real time traffic	Min: 1, Max: 15	8
	998	WTT entries in DMI reserved for high priority traffic	nDmiWttQoSRsv	Int	WTT entries in DMI reserved for high priority hard real time traffic	Min: 1, Max: 32	1
	999	RTT entries in DMI reserved for high priority traffic	nDmiRttQoSRsv	Int	RTT entries in DMI reserved for high priority hard real time traffic	Min: 1, Max: 32	1
	1102	Enable support for read response interleaving	useMemRspIntrlv	Bool			FALSE

Object	DocID	Title	Tcl	Type	Description	Allowed Vals	Default
	1103	Number of Address Translation Registers	nAddrTransRegisters	Int	Specifies the number of address translation registers that are available within DMI. Theses registers add capability to translate address on the AXI bus from DMI. Refer to Address Translation in the Reference Guide.	Min: 0, Max: 4	0
	1105	nWayPartitioningRegisters	nWayPartitioningRegisters	Int	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.	Min: 0, Max: 16	0
	1058	Enable Scratchpad	useScratchpad	Bool	This option enables scratch pad capability which is available only in DMI.		FALSE
	1056	Number of sets	nSets	Int	Specify the number of sets/ entries in the Cache.		16
	1057	Number of ways	nWays	Int	Specify the number of ways/ associativity of the cache.	Min: 2, Max: 16	2
	1058	Enable Scratchpad	useScratchpad	Bool	This option enables scratch pad capability which is available only in DMI.		FALSE
	1064	Primary Set Select Bits (Left is LSB)	PriSubDiagAddrBits	Int	Specify address bits to be used as primary set select bits (Left most bit is LSB).		
	1065	Tag Bank Select Bits	TagBankSelBits	Int	Specify tag bank select bit. This bit must be one of the bits from the primary select bits.		
	1066	Data Bank Select Bits	DataBankSelBits	Int	Specify data bank select bit. These bits must be from the primary select bits.		
Distributed Virtual Memory Management Engine (DVE)							
	339	Domain	domain	Hier Str	Clock SubDomain associated with this unit.		
	179	Number of performance counters	nPerfCounters	Int	Number of performance counters per Ncore unit.		4
Non Coherent Agent Interface Unit (NCAIU)							
	339	Domain	domain	Hier Str	Clock SubDomain associated with this unit.		
	519	Assigned SnoopFilter	snoopFilter	Hier Str	Specify the snoop filter associated with this AIU.		
	538	Enable Proxy Cache	hasProxyCache	Bool	This options enables Proxy Cache support. This is supported only with AXI interface.		FALSE

Object	DocID	Title	Tcl	Type	Description	Allowed Vals	Default
	1735	Tag Banks	nTagBanks	Int	Number of tag banks in the Proxy Cache		1
	1736	Data Banks	nDataBanks	Int	Number of data banks in the Proxy Cache		1
	1540	Cache Replacement Policy	cacheReplPolicy	Str	Cache Replacement Policy		RANDOM
	539	Outstanding transaction table Entries	nOttCtrlEntries	Int	Specify the maximum number of outstanding native transactions this AIU should support.	Min: 8, Max: 1024	32
	146	Select Bits (Left is LSB)	aPrimaryBits	Int	Bits that select the native port. They need to be as many as log2(number of native ports i.e. cardinality), they cannot be in the LSBs inside a cache line. (Left most bit is LSB.)		
	147	Number of performance counters	nPerfCounters	Int	Number of performance counters per Ncore unit.		4
	121	Number of latency counters	nLatencyCounters	Int	Number of latency counters in the Ncore unit.		16
	893	Processors ID bits	AxIdProcSelectBits	Int	specify processor ID bits.		
	1056	Number of sets	nSets	Int	Specify the number of sets/ entries in the Cache.		16
	1057	Number of ways	nWays	Int	Specify the number of ways/ associativity of the cache.	Min: 2, Max: 16	2
	1058	Enable Scratchpad	useScratchpad	Bool	This option enables scratch pad capability which is available only in DMI.		FALSE
	1064	Primary Set Select Bits (Left is LSB)	PriSubDiagAddrBits	Int	Specify address bits to be used as primary set select bits (Left most bit is LSB).		
	1065	Tag Bank Select Bits	TagBankSelBits	Int	Specify tag bank select bit. This bit must be one of the bits from the primary select bits.		
	1066	Data Bank Select Bits	DataBankSelBits	Int	Specify data bank select bit. These bits must be from the primary select bits.		
		Network					
	543	Units	components	Hier Str	Units		
	544	Maximum Packet Length (bytes)	maxPacketLength	Int	Maximum Packet Length (bytes)	Min: 1	1
	57	Packet width for Fixed Serial	defaultFixedPacketWidth	Int	Packet width for Fixed Serial		16
	79	Pipe adapter data fifo depth	defaultPipeAdapterDataFifoDepth	Int	Pipe adapter data fifo depth	Min: 0, Max: 1024	2

Object	DocID	Title	Tcl	Type	Description	Allowed Vals	Default
	80	Pipe adapter header fifo depth	defaultPipeAdapterHeaderFifoDepth	Int	Pipe adapter header fifo depth	Min: 0, Max: 1024	0
	81	ClockAdapter FIFO depth	defaultAsyncAdapterFifoDepth	Int	ClockAdapter FIFO depth		2
	83	Arbitration Policy	arbPolicy	Str	Arbitration Policy		RR1
	84	Switch input buffer depth	defaultSwitchInputBufDepth	Int	Switch input buffer depth	Min: 0, Max: 1024	2
	89	Switch output buffer depth	defaultSwitchOutputBufDepth	Int	Switch output buffer depth	Min: 0, Max: 1024	0
	625	Default Packet Serialization	defaultPacketSerialization	Str	This parameter defines the enum for the format of Packet Serialization		HEADER_PARALLEL
		Common Network Component Parameters					
	547	Domain	domain	Hier Str	Clock subdomain associated with this network		
		Pipe Adapter					
	565	Depth	depth	Int	Depth of the PipeAdapter	Min: 1, Max: 2	2
		Placeholder					
	636	RTL Prefix	wireRtlPrefix	Str	RTL Prefix. For a given block, all the ports must have the same writeRtlPrefix		placeholder
		Region					
	1297	Target Socket	physicalChannel	Hier Str	Region maps to one target (or an aggregate of multiple identical targets for address striping).		
	279	Base Address	memoryBase	Int	The base address of the region.	Min: 0	
	280	Size	memorySize	Int	Size of the memory. Each region size is power of 2, starting address is aligned to its size	Min: 0	
		Safety Configuration					
	397	Current ASIL Level	safetyConfig	Str	This parameter sets the safety mechanism for the current design		NO_ASIL
	398	Interconnect protection type	resiliencyProtectionType	Str	Inteconnect protection type. Both data and control header will be protected. Available options are: NONE: no pretection. PARITY: Error detection, parity protection. SECDED: Single bit error correction and double bit error detection, ECC protection.		NONE

Object	DocID	Title	Tcl	Type	Description	Allowed Vals	Default
	399	Memory Protection	memoryProtectionType	Str	Protection type for all memories in the Ncore system. Available options are: NONE : no pretection. PARITY : Error detection, parity protection. SECDED : Single bit error correction and double bit error detection, ECC protection.		NONE
	391	Resilience	resilienceEnabled	Bool	Enable resilience-related features in the Ncore system. Refer to the Reference Guide for details on available features		FALSE
	392	Unit duplication	duplicationEnabled	Bool	Enable unit duplication for all Ncore units only. Memories and interconnect logic are not duplicated; they may be protected separately.		FALSE
		Snoop Filter					
	91	Number of sets for the selected snoop filter	nSets	Int	Number of sets for the selected snoop filter. This value should be a power-of-2.	Min: 16, Max: 1048576	16
	92	Number of ways for the selected snoop filter	nWays	Int	Number of ways for the selected snoop filter.		4
	93	Number of victim buffer entries per DCE for the specified snoop filter	nVictimEntries	Int	The number of victim buffer entries for the tag filter.	Min: 0, Max: 64	2
	94	Set Select Bits (Left is LSB)	aPrimaryBits	Int	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]. (Left most bit is LSB.)		
		Subsystem					
	99	Subsystem Type	subsystemType	Str	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>

Object	DocID	Title	Tcl	Type	Description	Allowed Vals	Default
		Switch					
	53	Input buffer depth	inputBufferDepth	Int	Depth of buffers at each Switch Input Port. If the depth is 0, it is a Flow-through switch.		2
	7551	Port Data Width	portDataWidth	Int	Specify the data width of the ports of a switch. All ports have the same data width.	Min: 0, Max: 8192	0
	7554	Port Packet Serialization	portPacketSerialization	Str	Choice of packetization style for all ports of the switch. HEADER_PARALLEL: The first beat of a packet contains both header and data information and if subsequent data beats are needed, the following beats contain no header. This style requires the most wires but has the best latencies. HEADER_SERIAL: The header is sent in the first beat of the packet, and the subsequent beats carry data, if any. This style is a trade-off between wires and latency. FIXED_SERIAL: The packetization is similar to HEADER_SERIAL, but the beat width can be made narrower than what is necessary to send the header or data in a single cycle. This style saves wires, at the cost of requiring more beats to send the same information.		HEADER_PARALLEL
	96	Memory Type	memoryType	Str	This parameter is used to set the type of memory. This parameter is an enumerated type whose values are either FLOP or SRAM		FLOP
		Synthesis Settings					
	2043	Compile and Link Only	checkOnly	Bool	Whether to stop the RTL flow after compilation and linking, or to proceed to synthesis.		
	2044	Topographical Mode	topoMode	Bool	Whether to launch the synthesis tools in topographical mode, or WLM. The latter is faster but less accurate.		
	2045	Technology Node	techNode	Str	The technology node for the design.		CUSTOM
	2046	Clock Uncertainty	clockUncertainty	Int	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.	Min: 1, Max: 99	15

Object	DocID	Title	Tcl	Type	Description	Allowed Vals	Default
	2047	RTL Wrapper Dir	rtlWrapperDir	Str	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	Hard Macros	hardMacroDbs	Str	List of DB files containing compiled models of hard macros instantiated from RTL wrappers.		
	2049	Bottom-Up	bottomUpSynthesis	Bool	Characterize and compile major functional blocks (subdesigns) first, and then the toplevel.		TRUE
	2050	Max Transition	maxTransition	Int	Max signal transition time constraint, in ps.	Min: 0	150
	2051	Output Load	outputLoad	Int	Capacitive load for all primary outputs, in fF.	Min: 0	100000
	2052	Gate Area	gateArea	Int	Area of a reference gate, in um^2.	Min: 1	1
	2053	Gate Delay	gatePropagationDelay	Int	Reference gate propagation delay, in ps.	Min: 1	10
	2054	Wire Delay	wirePropagationDelay	Int	Wire propagation delay, in ps/mm (= ns/m).	Min: 1	1000
	2055	Max Wire Density	maxWireDensity	Int	Maximum wire density in a channel, in wires/mm (= kwires/m).	Min: 1	1000
	2056	Subdesign Instances	subDesignInstances	Str	List of instance names or whose modules should be synthesized separately as subdesigns. Can include patterns. Only applicable in bottom-up synthesis flow.		
	2057	Incremental Optimization	incrementalOpt	Bool	Enables DC to perform incremental optimization. This works only on mapped logic.		TRUE
	2058	Ulvt Percentage	ulvtPercentage	Int	% of ulvt cells can be used by the tool.	Min: 0	0
	2059	Compile command	compileCommand	Str	DC mapping and optimization command.		compile_ultra -spg -no_boundary_optimization -gate_clock
		Width Adapter					
	1229	Pipeline	pipeline	Bool	Whether the WidthAdapter has a pipeline stage or transfers flits combinationally.		FALSE
	14	Input Port Data Width	inputPortDataWidth	Int	Derived data width from input port.	Min: 0, Max: 8192	0

Object	DocID	Title	Tcl	Type	Description	Allowed Vals	Default
	15	Output Port Data Width	outputPortDataWidth	Int	Derived data width from output port.	Min: 0, Max: 8192	0
	617	Suppress Blank Beats	suppressBlankBeats	Bool	U: false, F: true, T: true		FALSE
		Boot Info					
	1	Boot Region Home Unit ID	regionHui	Int	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	Boot Region Size	regionSize	Int	specifies the size of boot region. Minimum is 4KB and must be a power of two.		4096
	3	Boot Region Home Unit Type	regionHut	Str	Specify the home unit Type associated with the boot region. This can be either DMI (Memory) or DII (IO device).		DMI
		Topology					
	407	Number of DVM command request credits	nDvmCmdCredits	Int	Maximum number of DVM command requests an AIU can have in flight.	Min: 1, Max: 4	1
	408	Number of DVM snoop request credits	nDvmSnpCredits	Int	Maximum number of DVM snoop requests an AIU can have in flight on its native interface.		2
	395	Template	coherentTemplate	Str	Control and data network options: TwoCtrlOneDataTemplate: Adds support for two control and a single data network. ThreeCtrlOneDataTemplate: Adds support for three control and a single data network. Refer to the Reference Guide.		
	393	Native interface protection	nativeIntfProtEnabled	Bool	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
	396	Duplicate unit delay	interUnitDelay	Int	Delay between functional unit and delay unit. Delay can be specified in number of clock cycles.	Min: 1, Max: 4	1
	389	General purpose address regions	nGPRA	Int	Number of general purpose address regions that the system can support. Refer to the Reference Guide.	Min: 1, Max: 24	1

Object	DocID	Title	Tcl	Type	Description	Allowed Vals	Default
	406	TimeOut threshold	timeOutThreshold	Int	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.	Min: 1, Max: 2147483647	16384
	367	Enable QoS support	qosEnabled	Bool	Enable QoS capability.		FALSE
	368	QoS value to priority map	qosMap	Str	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	QoS event threshold	qosEventThreshold	Int	QoS starvation threshold. Maximum number of high priority requests that can bypass a lower priority request.	Min: 1, Max: 8192	16
	177	Number of trace entries in the buffer	nMainTraceBufSize	Int	Specifies the number of trace entries in the buffer.	Min: 32, Max: 1024	64
	178	Number of Trace Registers	nTraceRegisters	Int	Specifies the number of trace register sets present in the system per AIU.	Min: 1, Max: 4	1
	388	Depth of Synchronizers	syncDepth	Int	Specifies the number of synchronization registers at an async clock crossing boundary	Min: 2, Max: 8	2

11 Glossary

Table 10. Terms and Abbreviations

Terms	Descriptions
Active State	When a component is tracking state pertaining to outstanding transaction.
Active Transaction Table	Entry to track coherent protocol transactions.
ATU	Arteris Transport Unit used in Non-Coherent Transport Interconnect.
BIST	Built-In-Self Test
BR	Boot Region
Cache Coherence	Consistency of data in multiple caches.
Cache Lines	Data allocated in caches in quantities which are larger than the typical size of memory access performed by a processing element.
Cache State Models	Consists of MSI-Invalid model and MEI model.
Caching Agents	Fully-coherent agents which incorporate caches that store coherent copies of data and implement snooping interfaces on which coherence operations are presented.
Coherent Agent Interface	Allows coherent agent access to both the coherent and non-coherent subsystem.
CAIU	Coherent Agent Interface Unit. Can be CHI-A, CHI-B, or ACE. CHI 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.
CDTI	Control and Data Transport Interconnect
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
CMD	Command Message
CNI	Coherent Network Interface, used internally in Maestro.
ConfigSocket	Socket used for the Configuration Networks
CSTI	Control and Status Transport Interconnect
Database	The database contains all the information entered by the user, as well as information regarding the state of the tool, for a project.

Table 10. Terms and Abbreviations

Terms	Descriptions
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).
DTR	Data Read Message
DTW	Data Write Message
DVE	Distributed Virtual Memory Management Engine. It 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.
ERG	Exclusive Reservation Granule
Dynamic Memory Group	Container for DMI sockets. It can hold maximum of 4 DMI sockets.
Fabric	A Fabric is a set of interconnected packets switching and transport hardware units.
Generator	Engine in Maestro used to carry out a functionality.
GPAS	General Purpose Address Space
GRB	Global Register Block. It manages memory address.
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 master 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.
IPG	The IP Generator 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.
Master Agent	Functional logic block that initiates read and write transactions to a particular physical address.
MEI Model	In this model, the cache implements the modified, exclusive, and invalid cache states.
Memory Agent	Provides access to memory locations that behave like “normal” memory, and accesses to physical addresses associated with a memory agent.
Memory Set	Set of Memory Groups. The memory groups in the set must include all the DMIs.
MIG	Memory Interleave Group
MPF	Maestro Project files
MRD	Memory Read Message
MSI-Invalid Model	In this model, the cache implements the modified, shared, and invalid cache states.

Table 10. Terms and Abbreviations

Terms	Descriptions
Native Interface	An external interface that communicates directly with a master agent or a slave agent.
NCAIU	Non-Coherent Agent Interface Unit
NCNI	Non-Coherent Network Interface
NCTI	Non-coherent transport interconnects are a non-coherent subsystem that only supports non-coherent access semantics.
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 is as a class of communication fabrics. Arteris IP interconnects are based on NoC designs.
Non-coherent Bridges	Translate non-coherent transactions from the non-coherent subsystem into coherent transactions to the coherent subsystem.
Non-coherent Transaction	Directly accesses the data at the corresponding memory location.
Non-memory Agent	Provides access to functionality that behaves like device functionality, and accesses to physical addresses associated with a non-memory agent.
NRS	Ncore Register Space
NRU	Not-recently-used policy tracks cachelines that have been recently used.
Null Filter	Stores no information about its associated caching agents, and for every coherent transaction, the directory broadcasts snoop messages to those caching agents.
Object	An 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.
OTT	Outstanding transaction table tracks the outstanding protocol transactions as they progress through the various phases of the coherence protocol.
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).
POS	Point of Serialization.
Port	A Port is 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
Presence Vector Filter	Each caching agent is represented by a bit that indicates whether the caching agent possesses a valid copy of the cache line.
Project	A project is a container of all information required to create a Communication Subsystem. All elements of a projects are related to each other. Maestro let the user work on one project at a time.
Protocol Control Messages	Transmit coherence information between a pair of components.
Protocol Data Messages	Transfers data between a pair of components.
Proxy Cache	Caches data on behalf of the non-coherent agents that have accessed the bridge.
Random Policy	Uses a pseudo-random pointer to choose a valid cache line for replacement.
RB	Request Buffer

Table 10. Terms and Abbreviations

Terms	Descriptions
Route	Route is an ordered list of Transport units where traffic can go through from a source 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 and the address decode and transformation function used to route requests between them.
RPN	Register Page Number
RTT	Read transaction table implements storage for the read transaction state and maybe configured with data buffers for storing read response data from the memory agent
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.
SF	Snoop Filter
SMC	System Memory Cache
SMI Port	Symphony Message Interface Port
Snoop Filters	Track the cache state of caching agents.
SNP	Snoop Message
SoC	A class of Integrated Circuits built by assembling multiple components (CPU, memory, accelerators, etc.) around a Communication Subsystem.
Socket	A Socket is 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.
STR	State Reply Message
Subsystem	A Subsystem is any combination of ArterisIP products including 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.
Tag Filter	Set associative structure that tags entries by physical address and stores cache line validity and optionally, ownership information about its associated caching agent.
Temporal Locality	Dimension where allocating frequently accessed data in a cache decreases the time that a processing element may spend waiting for data.
Transport Interconnects	Communication among components occurs via this interconnect.
Uncorrectable Errors	Double-bit errors with SECDED protection
UPD	Update Message
VC	Virtual Channel
Victim Cache	Buffers data evicted from the caching agent caches in addition to speculative memory read data.
Write Transaction Table	Implements storage for memory write transaction state

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