Building and Analyzing On-Chip Networks using CHAIN® architect

(CHAINworks 2.1.1)

License

Disclaimer

Table of Contents

[THIS PAGE INTENTIONALLY LEFT BLANK]

Building and Analyzing On-Chip Networks using CHAIN® architect

Introduction

This guide describes how to use the Silistix® CHAIN®architect software, in conjunction with the Silistix Connection Specification Language (CSL™) to build and analyze a network on a chip (NoC). This guide uses screen shots and methods from the Linux® version of CHAINarchitect. However, the Microsoft® Windows® version is similar.

With CHAINarchitect, you can …

- Specify systems in CSL using the integrated, syntax-aware text editor.
- Check the CSL specification for syntax or connection errors.
- Graphically view the resulting on-chip network topology, including latency and bandwidth.
- View the statistics for the generated network, including highly-accurate predictions for …
  - Silicon area for the targeted fabrication vendor and process node
  - Power consumption for the network
  - Latency and bandwidth for connections between endpoints on the network
- Generate structural Verilog models for the various interface protocol adapters and network connections used in the design. Use these netlist files to build your System-on-a-Chip (SoC).
- Generate Verilog and/or SystemC simulation models and testbenches, including command scripts.
- Generate script files for static timing analysis (STA) of the generated network.
- Generate test patterns for popular automated test equipment (ATE).

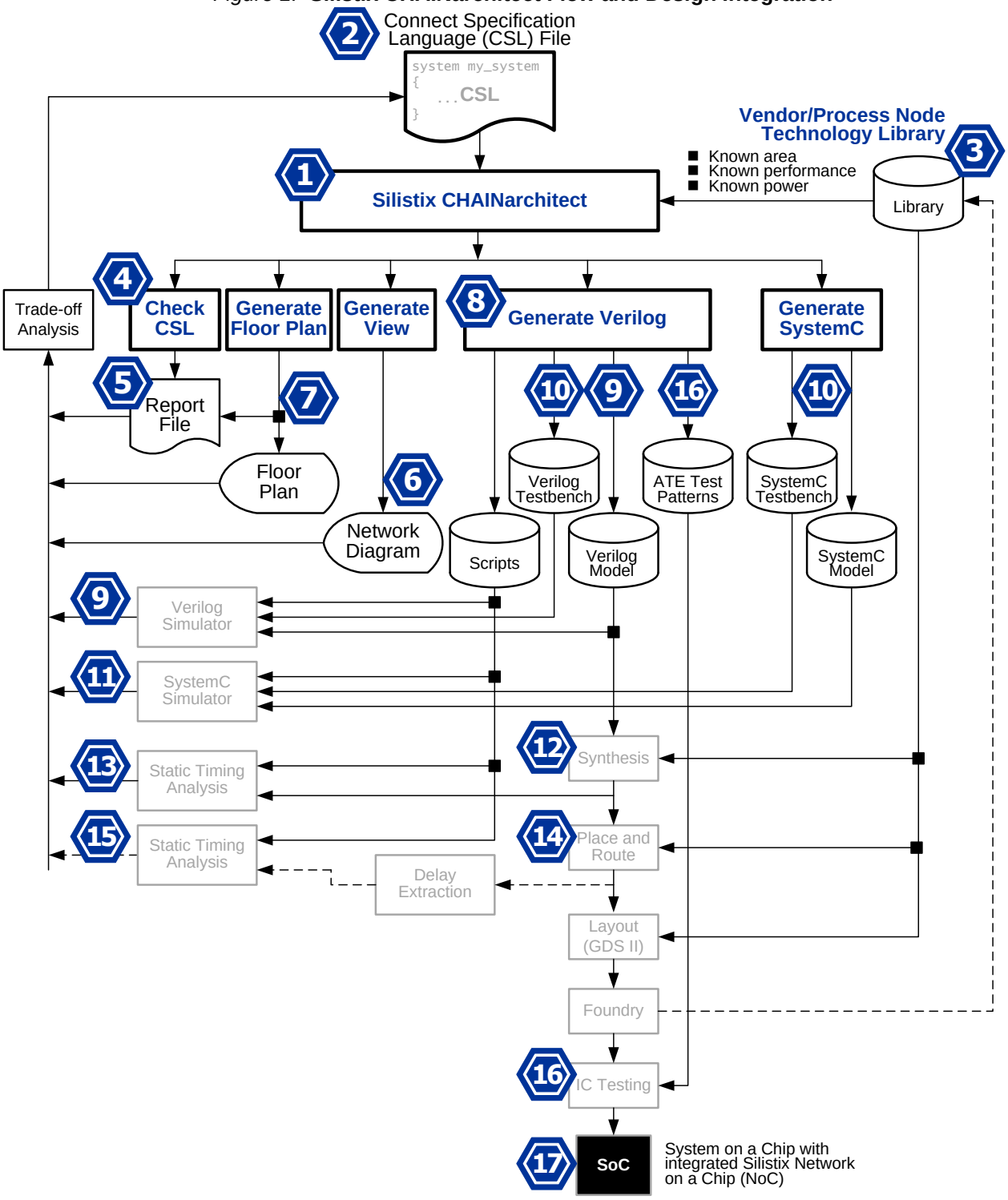
CHAINarchitect Overview

Figure 1 depicts the complete Silistix CHAINarchitect design flow, starting from design specification using the Connection Specification Language (CSL) all the way to the final, tested System-on-a-Chip (SoC) device that integrates the proven CHAIN network. CHAINarchitect allows you to quickly create and analyze possible network implementations so that you can find the best solution for your specific application.

1. **CHAINarchitect** is the architect's "design cockpit" to create, analyze, and integrate a Silistix CHAIN network into a working system-on-a-chip application.

2. The architect integrates the multiple IP cores in the SoC design by describing the connections and traffic characteristics of those connections. The Silistix Connection Specification Language (CSL) is designed specifically for this purpose. See "***Describing a System in Connection Specification Language (CSL)***."

Figure 1: **Silistix CHAINarchitect Flow and Design Integration**



3. The technology library is a key aspect of the proven Silistix network. The library contains adapters to popular IP interface protocols and pre-implemented, pre-verified, pre-characterized hard macro blocks to generate a purpose-built on-chip network to connect the system blocks. The pre-implemented hard macros also provide highly-accurate area, timing, and power numbers, allowing you to quickly analyze and iterate your design, all without ever running place and route or proceeding to layout.

4. Analyze the CSL file for syntax errors and generate a report summarizing network characteristics based on the CSL specification. See "Check the CSL File, Generate a Report File" on page 14.

5. Begin analyzing various implementation trade-offs using the information contained in the report file, including the resulting area, latency, power, and bandwidth for the CHAIN network. Tune the overall system performance. The clockless CHAIN network allows each of your IP blocks to operate at their optimal frequency. No longer is your design constrained to a magic frequency, common to all the IP blocks. See "View and Interpret the CSL Compiler Report File" on page 17.

6. The First Placement Estimator (FPE) generates a floor plan for the system. By considering placement effects, CHAINarchitect automatically inserts pipelatch components to re-buffer network signals. The FPE tool also updates the report file and optionally generates a DEF placement file. See "Generate First Placement Estimation" on page 29.

7. To help visualize the resulting network, CHAINarchitect generates a network diagram from the CSL file. The diagram shows the endpoints on the network, network adapters, connections along the network, and statistics about each connection. See "View a Generated Network" on page 22.

8. Once you are satisfied with your CHAIN network design, then generate Verilog code. See "Generate Verilog Structural Netlist and Validation Models" on page 36.

9. The Verilog structural netlist contains synthesizable code to connect your IP blocks to the CHAIN network gateways. Similarly, the generated Verilog code instantiates the hard macro functions that build the CHAIN network fabric.

10. CHAINarchitect also generates Verilog validation models, including testbenches and stimulus files. Use these files with industry-standard Verilog simulators for more detailed analysis of your Network-on-Chip.

11. As an alternate means to validate and analyze CHAIN networks, CHAINarchitect also generates a SystemC model along with testbenches and stimulus files. See "Generate SystemC Validation Models" on page 37.

12. Synthesize the CHAIN network with the remainder of your SoC design using the structural netlist provided in Step 8 above.

13. After synthesis, use a Static Timing Analyzer (STA) and the scripts generated by CHAINarchitect to further analyze the performance your SoC design with the integrated CHAIN network. By its very design, the CHAIN network makes timing closure easier and faster than using typical synchronous bus designs.

14. If the SoC design and CHAIN network meet your application requirements, proceed to place and route. The place-and-route tools integrate the Silistix network hard macros and wire length information, guaranteeing network performance and area characteristics.

15. Perform static timing analysis after place-and-route using extracted delays for even higher confidence in the final system-level design.

16. CHAINarchitect generates test patterns for popular automated test equipment. After your SoC devices return from the fab, use these patterns to test the CHAIN network.

17. Congratulations! Your SoC design now includes your time-saving, solution-optimized, low-cost, royalty-free, power-saving Silistix CHAIN network.

Invoke CHAINarchitect

The CHAINarchitect software executes on either Linux or Windows computers. The FLEXnet® license manager must be running either on the local computer or on a server somewhere on the network.

Linux

After installing the CHAINworks software and setting up the runtime environment, simply invoke CHAINarchitect from a terminal window.

If CHAINworks is in your path environment, simply type …

```
CHAINarchitect
```

If CHAINworks is not in your path, then type …

```
$SILISTIX_HOME/bin/CHAINarchitect
```

If the CHAINworks software is installed correctly, the CHAINarchitect startup screen should appear.

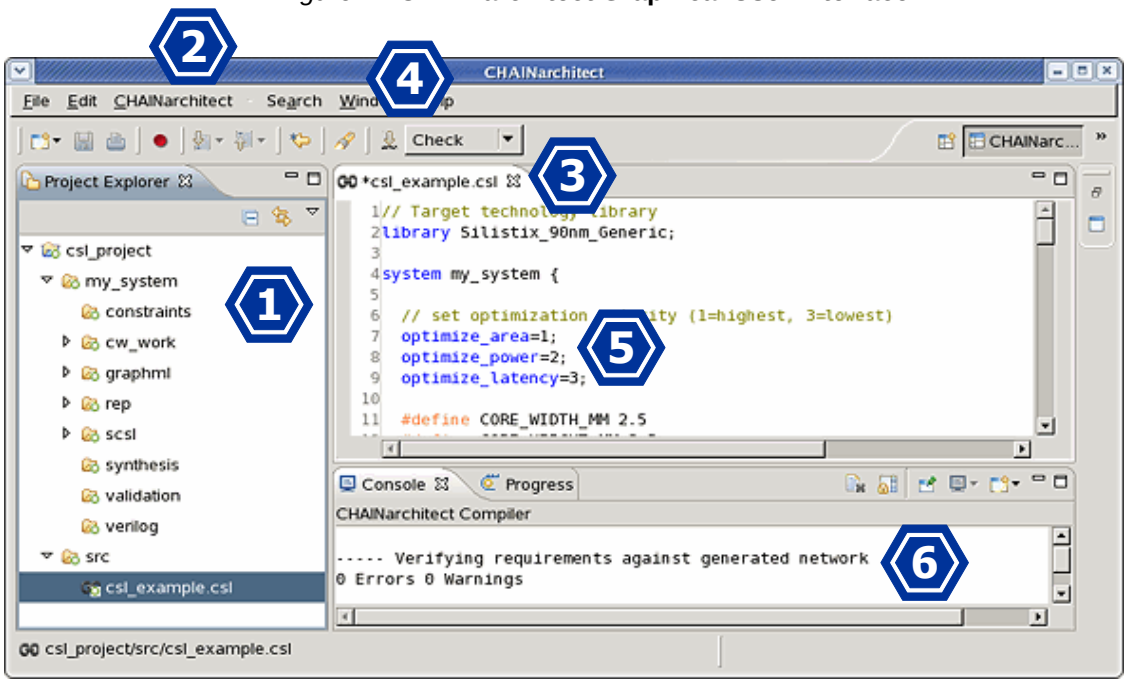
Windows

After installing the CHAINworks software, double-click the CHAINarchitect icon installed on the desktop, or select **Start → CHAINarchitect → CHAINarchitect** from the Windows Start menu.

CHAINarchitect Graphical Interface Overview

Silistix CHAINarchitect provides a graphical interface with integrated project management. The CHAINarchitect interface appears in Figure 2.

Figure 2: **CHAINarchitect Graphical User Interface**



1. The **CHAINarchitect Explorer window** represents entire current CHAINarchitect project, showing the source and result files that are part of the project.
2. Use the **CHAINarchitect Menu** to invoke various CHAINworks tools to process a design described in CSL. The CSL must be selected in the main windows.
3. Source and result files appear in the **Main Window**. Each open file appears as a tabbed window. The CSL source file tab must be selected in order to process the file.
4. The Go Button and current button setting. See page 41 for more information.
5. Keywords within source and result files are color coded according to type.
6. The **Console Window** displays error and warning messages.

Create a New CHAINarchitect Project

To create a new project, click **File → New → CHAINarchitect Project.**, as shown in Figure 3.

Figure 3: **Create a New CHAINarchitect Project**

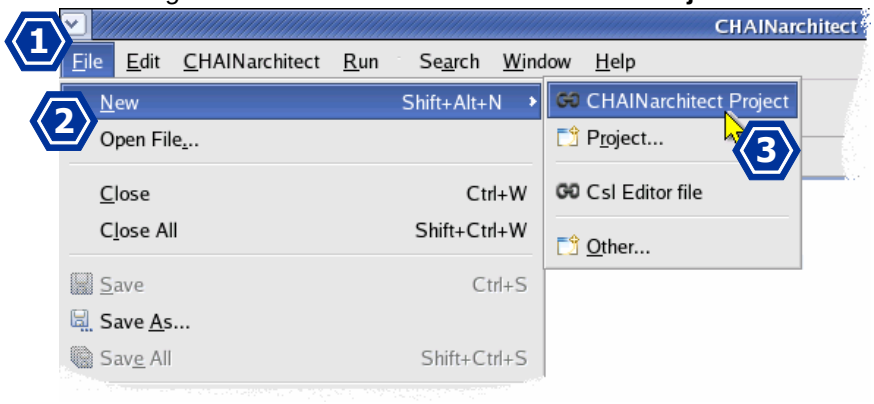
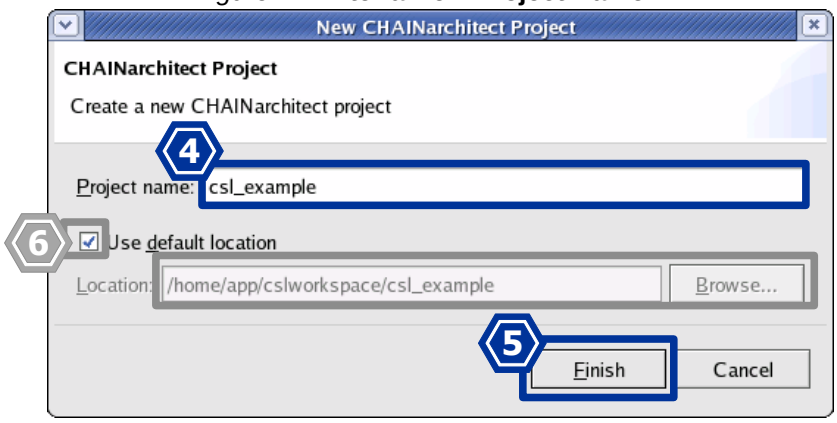


Figure 4: **Enter a New Project Name**



Enter the **Project Name** as shown in Figure 4.

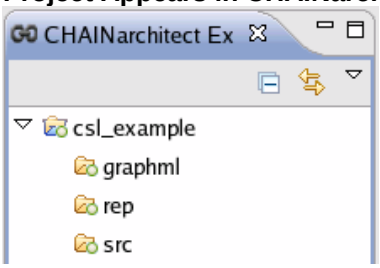
Click **Finish**.

OPTIONAL: By default, the new project is created in your home directory, under the `cslworkspace` subdirectory under your home directory or in `$HOME/cslworkspace/`. If you prefer a different location, uncheck **Use default location**, then click **Browse** to choose a new location.

The new project then appears in the CHAINarchitect Explorer window, as shown in Figure 5. Three new subdirectories are created.

- `graphml` contains a graphical representation of the finished CSL application.
- `rep` contains report files generated by CHAINarchitect.
- `src` contains the CSL source file.

Figure 5: **New Project Appears in CHAINarchitect Explorer**



The project itself is currently empty. At this point, either …

- Create a New CSL File, or
- Import an Existing CSL File

Create a New CSL File

To create a new project, click **File → New → Csl Editor File**. Specify the name of the new file. CHAINarchitect then creates the file, which initially contains the CSL template file.

Import an Existing CSL File

To import an existing CSL file into the current project, follow these steps.

1. As shown in Figure 6, right-click on `src` in the CHAINarchitect Explorer window.

2. Select **Import** from the pop-up menu.

Figure 6: **Import CSL Source File**

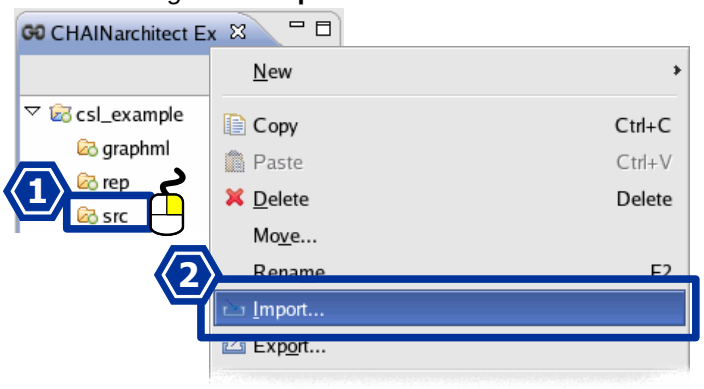
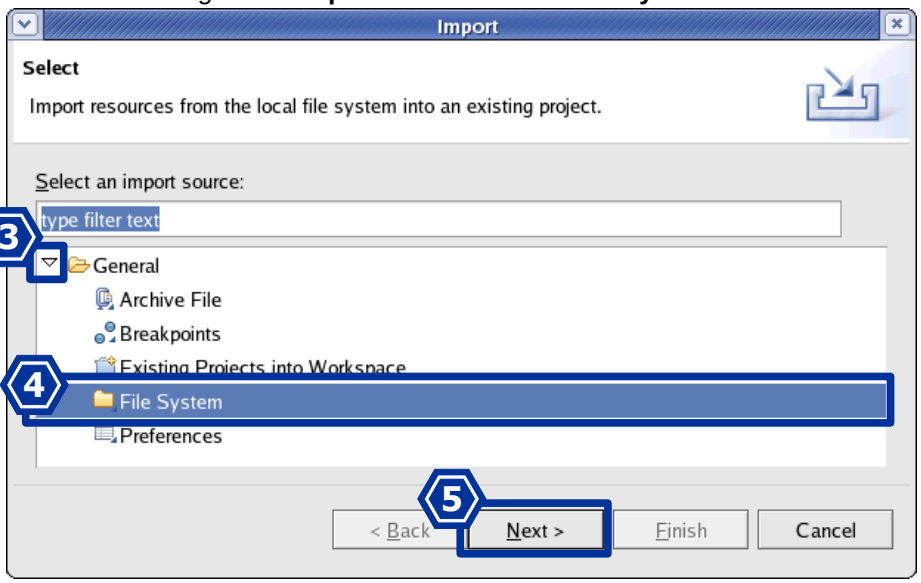


Figure 7: **Import CSL File from File System**

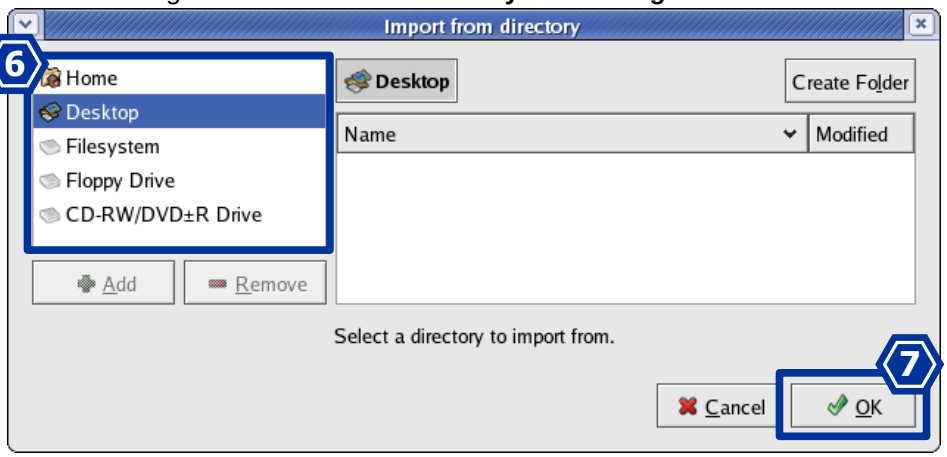


3 As shown in Figure 7, expand **General**.

4 Click **File System**.

5 Click **Next**.

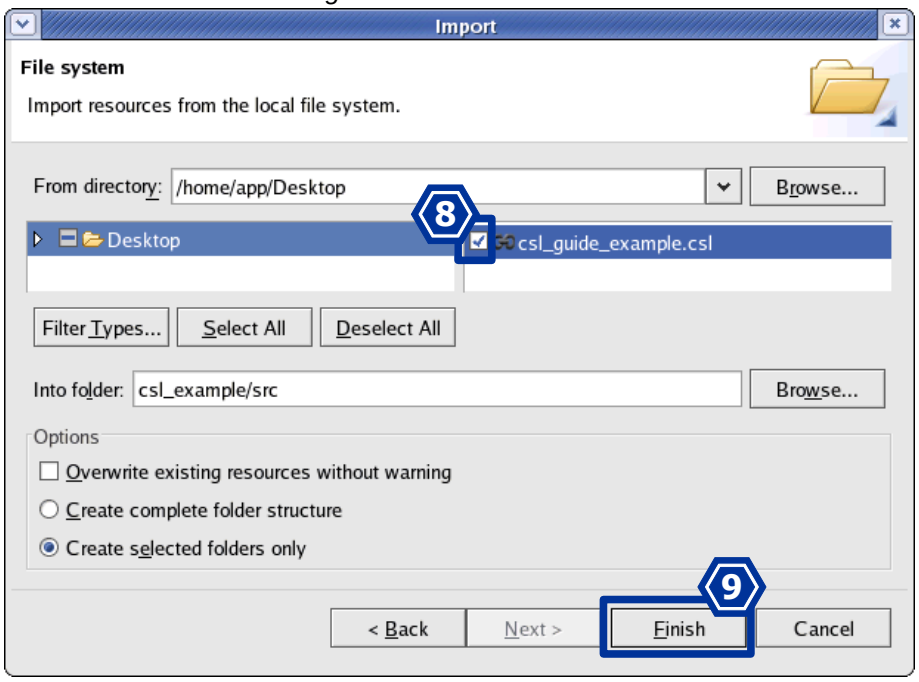
Figure 8: **Browse to Directory Containing CSL File**



6 As shown in Figure 8, browse to the directory containing the CSL file that you wish to import.

7 Click **OK**.

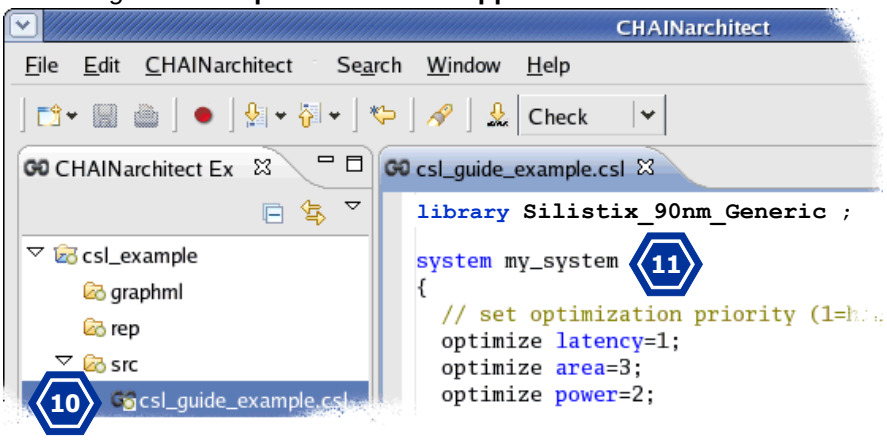
Figure 9: **Select CSL File**



8 As shown in Figure 9, check the option box next to the CSL file.

9 Click **Finish**.

Figure 10: **Imported CSL File Appears in CHAINarchitect**



10 As shown in Figure 10, the name of the imported CSL file now appears under the **`src`** subdirectory in the CHAINarchitect Explorer window.

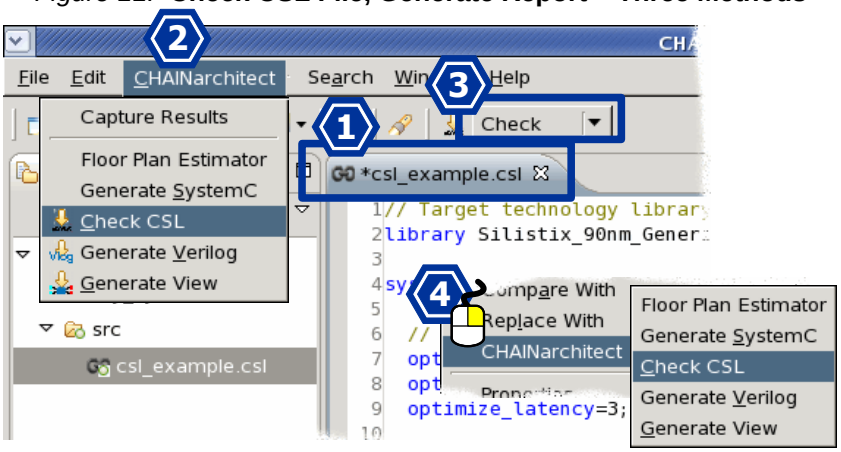
11 The content of the CSL file appears in the main window area.

Check the CSL File, Generate a Report File

In CHAINarchitect, there are three different methods to check the syntax and settings of a CSL file. If the CSL file is relatively free of syntax or other errors, then CHAINarchitect also generates a report file.

1. To check a CSL file or generate other output files, you **must first click the CSL file tab** in the main window, as shown in Figure 11.

Figure 11: **Check CSL File, Generate Report – Three Methods**



After selecting the CSL file, there are three possible ways to check the file, as listed below.

2. **Method A:** From the main menu, click **CHAINarchitect → Check CSL**.

3. **Method B:** Set the Go Button to **Check**, then click the Go button,  .

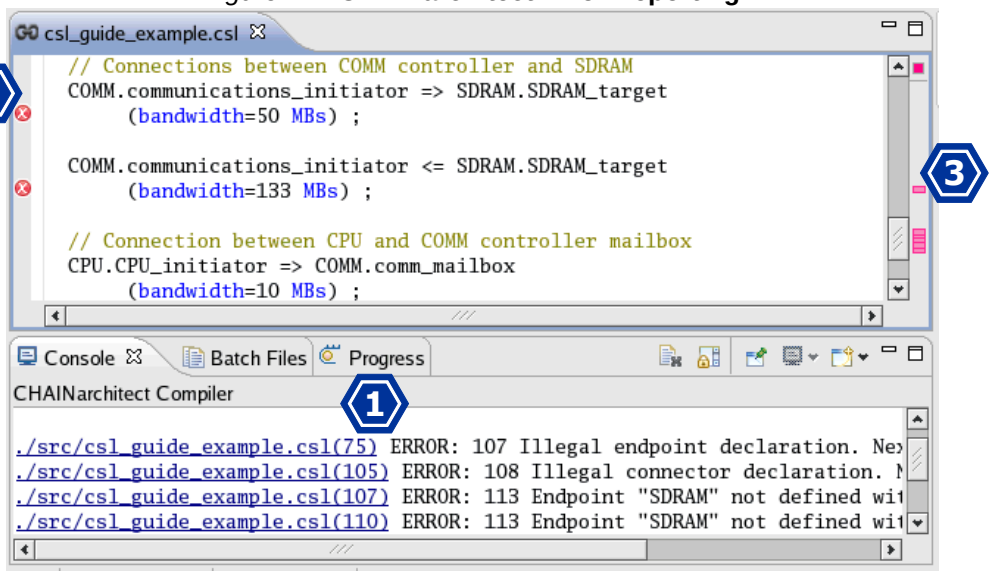
4. **Method C:** From within the CSL file, **right-click** and then select **CHAINarchitect → Check CSL** from the pop-up menu.

Error Reporting

CHAINarchitect has extensive CSL error checking, both for syntax errors and for connectivity or specification errors. Figure 12 illustrates how errors are generally presented in the graphical interface.

View Errors

Figure 12: **CHAINarchitect Error Reporting**



1. As shown in Figure 12, CHAINarchitect reports errors in the Console window. Generally, errors are hyperlinked to locations in the CSL file displayed above. Click a hyperlink to jump to the associated line.

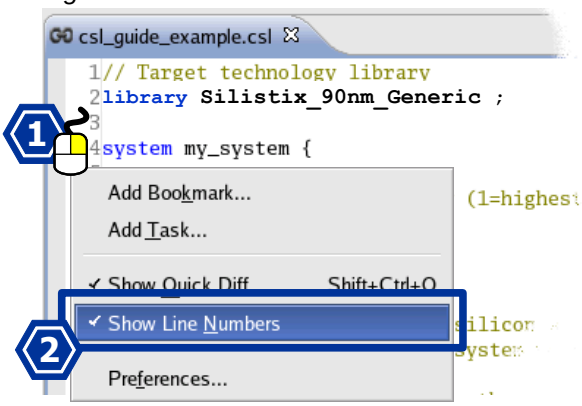
2. To quickly locate potential errors in text that is currently displayed, look for the red 'x' circles (⊗) along the left edge of the CSL file. Click a red circle to jump directly to the associated line.

3. The red squares along the right edge of the display indicate the relative position of errors for the entire file, including text that is currently vertically scrolled outside the display window. Click a red square to scroll to the associated text.

Turn On/Turn Off Line Numbers

CHAINarchitect typically reports errors along with their associated line numbers. Generally, you can jump directly to the problem line by clicking on the link provided by CHAINarchitect. However, follow these steps to turn on or turn off line numbers in the CSL file.

Figure 13: **Turn On or Off Line Numbers**



1. As shown in Figure 13, right-click on the vertical gray bar along the left edge of the text window.
2. From the resulting pop-up window, check or uncheck **Show Line Numbers**.

Change Editor Keyword Highlighting Colors

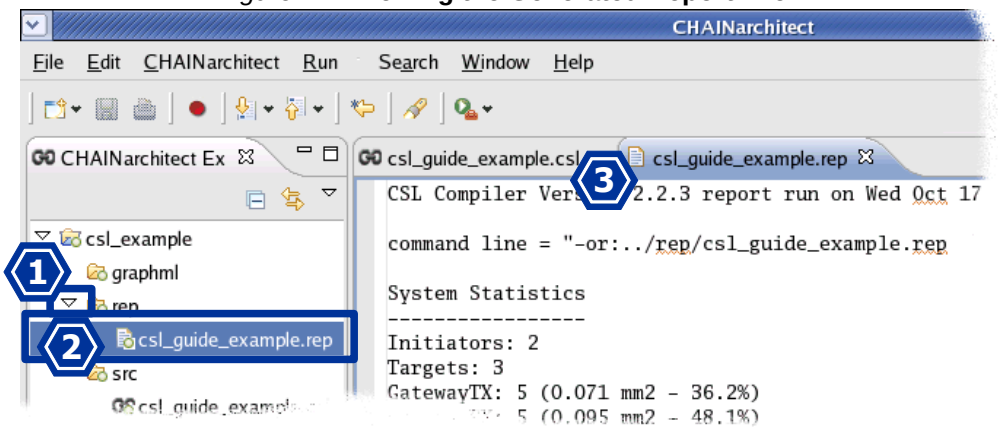
To change the editor settings, click **Window → Preferences** from the CHAINarchitect menu, expand **CHAINarchitect Preferences** and select **Csl Editor Preferences**.

View and Interpret the CSL Compiler Report File

View the Report File

The report file is generated automatically when you Check the CSL File, Generate a Report File. To view the report, follow the steps outlined below.

Figure 14: **Viewing the Generated Report File**



1. As shown in Figure 14, expand the `rep` subdirectory in the CHAINarchitect Explorer window.
2. Double-click the underlying `*.rep` file.
3. The report file appears as a tabbed editor window.

Read and Interpret the Report File

The report file has multiple sections, as listed in Table 1.

Table 1: **Report File Sections**

Section	Description
System Statistics	Summarizes the resulting network, showing number of resources required, area, power, and aggregate bandwidth.
Network Bill of Materials	Lists the specific Silistix library elements used to build the network and their associated silicon area.
Roundtrip Connections	Lists all endpoint-to-endpoint roundtrip connections on the network and relevant latency and bandwidth results.
Violations	Reports any violations and any differences between the requested characteristics and those that CHAINarchitect could deliver.

System Statistics

The System Statics section of the report file provides a general overview of the generated Silistix network topology. As shown below, the report lists the various resources required to build the network, the silicon area associated with each function, and the associated percent fraction of the area used. If the `area` statement was specified in the CSL source file, then the percentage value reflects how much silicon area of the entire design is required to implement the particular feature. If

the `area` statement was not specified, then the percentage value is the fraction of the silicon area consumed by this feature when compare to just the area used by the Silistix CHAIN network.

Similarly, if the `power` statement was specified in the CSL source file, then the system energy reported is the total power for the system, plus the typically small additional energy required by the Silistix CHAIN network.

Figure 15 shows an example of the system statistics reported.

- The number of Adaptors on the network, as described in the CSL source file, and their resulting combined silicon area.
- The number of transmit (TX) and receive (RX) gateways on the network, and their resulting silicon area.
- The number or Routes, Merges, Switches, Serdes (serializer/deserializer), and FIFOs required to connect the network, and their resulting silicon area.
- The total silicon area required to implement the Silistix network
- The active power consumed by the Silistix network

Figure 15: **Example System Statistics from Report File**

```
=================
SYSTEM STATISTICS
=================

                                          Active
Component       count   area     area    Power
                        (mm2)   (kgates) (mW)
Adaptor+CGP         5    0.380    83.28  8.576
TX                  5    0.793   173.70  6.119
RX                  5    1.025   224.58  6.119
Route               1    0.066    14.38  1.125
Merge               1    0.038     8.42  0.000
Switch              2    0.331    72.60  2.229
Serdes              0    0.000     0.00  0.000
Fifo                0    0.000     0.00  0.000
Pipelatch           0    0.000     0.00  0.000
----------
Soft IP                  0.380    83.28  8.576
Hard IP                  2.254   493.68 15.591
User IP                  0.000     0.00  0.000
       Total             2.634   576.95 24.167
```

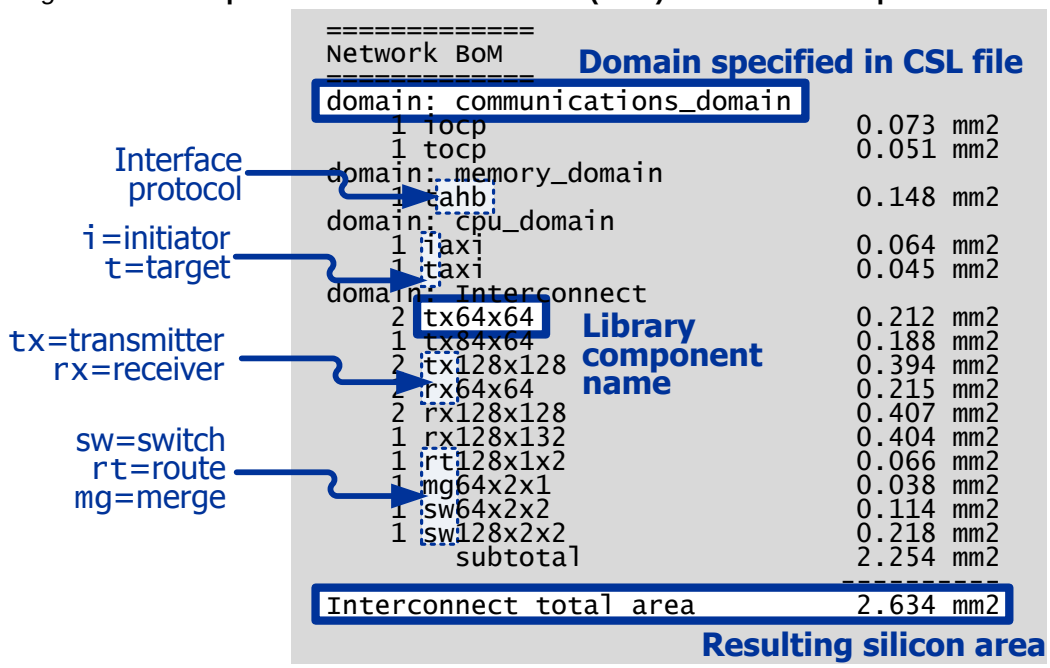
Network Bill of Materials (BoM)

The Network Bill of Materials section of the report file lists every Silistix library component used to build the network.

Figure 16 shows an example snippet of the reported bill of materials.

- The report is organized by each domain.
- The report lists the number and name of library components used within the domain, along with their associated silicon area for the target technology library.
- The total silicon area is also reported, which matches the total silicon area also reported in the System Statistics. However, the Network Bill of Materials provides a finer level of detail.

Figure 16: **Example Network Bill of Materials (BoM) Section from Report File**




```
=============
Network BoM
=============
domain: communications_domain
    1 iocp                          0.073 mm2
    1 tocp                          0.051 mm2
domain: memory_domain
    1 tahb                          0.148 mm2
domain: cpu_domain
    1 iaxi                          0.064 mm2
    1 taxi                          0.045 mm2
domain: Interconnect
    2 tx64x64                       0.212 mm2
    1 tx84x64                       0.188 mm2
    2 tx128x128                     0.394 mm2
    2 rx64x64                       0.215 mm2
    2 rx128x128                     0.407 mm2
    1 rx128x132                     0.404 mm2
    1 rt128x1x2                     0.066 mm2
    1 mg64x2x1                      0.038 mm2
    1 sw64x2x2                      0.114 mm2
    1 sw128x2x2                     0.218 mm2
      subtotal                      2.254 mm2
                                  ----------
Interconnect total area             2.634 mm2
```

Roundtrip Connections

The Roundtrip Connections section of the report file lists the connection paths on the network, as defined in the CSL file.

Figure 17 shows an example snippet from the connections section of the report file.

- The report is organized by each endpoint-to-endpoint connection on the network, all in a single direction.
- The `outstanding`, `burstsize`, `latency`, and `bandwidth` characteristics requested for the port, as specified in the CSL source file, are listed.
- The optimization settings for the connection, from the CSL source file are listed in priority order.
- The `cmd path` line shows the entire network command path, from the initiator at the sending endpoint, through any intermediate connections, to the target endpoint at the other end. The line also indicates the specific Silistix library elements used in the network command path. See Figure 18 for a diagram of the command path.
- The `rsp path` line shows the entire network response path, from the target endpoint, through any intermediate connections, back to the initiator endpoint at the other end. The line also indicates the specific Silistix library elements used in the network response path. See Figure 18 for a diagram of the response path.
- The `Sustained bandwidth` describes the maximum bandwidth sustained over the specified network, along with any slack or excess bandwidth above and beyond that specified in the CSL file.
- The `Interconnect roundtrip latency` is the sum of the network delays through the command and response paths, as shown in Figure 18. This value only includes any network transit delays, and does not include any delays or latency through the IP endpoint blocks.

- The `System roundtrip latency`, shown in Figure 18, is the sum of the network roundtrip latency listed above, plus any `write_reponse` or `read_response` delays specified in the source CSL file.
- The `Dynamic power per transfer` lists the amount of power consumed by each packet transferred over the network, measured in microwatts (µW).
- The `Max utilization` represents the loading on this particular connection. The number represents the fraction of full loading to achieve the requested bandwidth. Low numbers mean that the connection can potentially carry more bandwidth. This line also indicates which library component is the bottleneck in the network path, carrying the highest amount of traffic. In this example, it is the library element named `mg32x2x1_68`, which is midway through the response path in the example shown in Figure 17.

Figure 17: **Example Roundtrip Connections Section from the Report File**

```
====================
Roundtrip Connections
====================

Mode: my_system
---------------
cpu_domain.CPU.CPU_initiator (400.0MHz) => memory_domain.SDRAM.SDRAM_target (133.0MHz)
  (req: outstanding=8, burst=256 bits, optimize by: latency, power, area)
  cmd path = {iaxi_0 -> tx128x128_62 -> rt128x1x2_63 ->
              sw128x2x2_64 -> rx128x128_65 -> tahb_2}
  rsp path = {tahb_2 -> tx96x32_66 -> sw32x2x2_67 ->
              mg32x2x1_68 -> rx32x64_69 -> iaxi_0}
  Sustained bandwidth (1328.000 Mbs)  slack: 1315.896 Mbs
  Interconnect roundtrip latency: 84.503 ns
  System roundtrip latency (90.000 ns)  slack: -19.503 ns
  Dynamic power per transfer: 364.000 uw
  Max utilization 18.250% at mg32x2x1_68
```

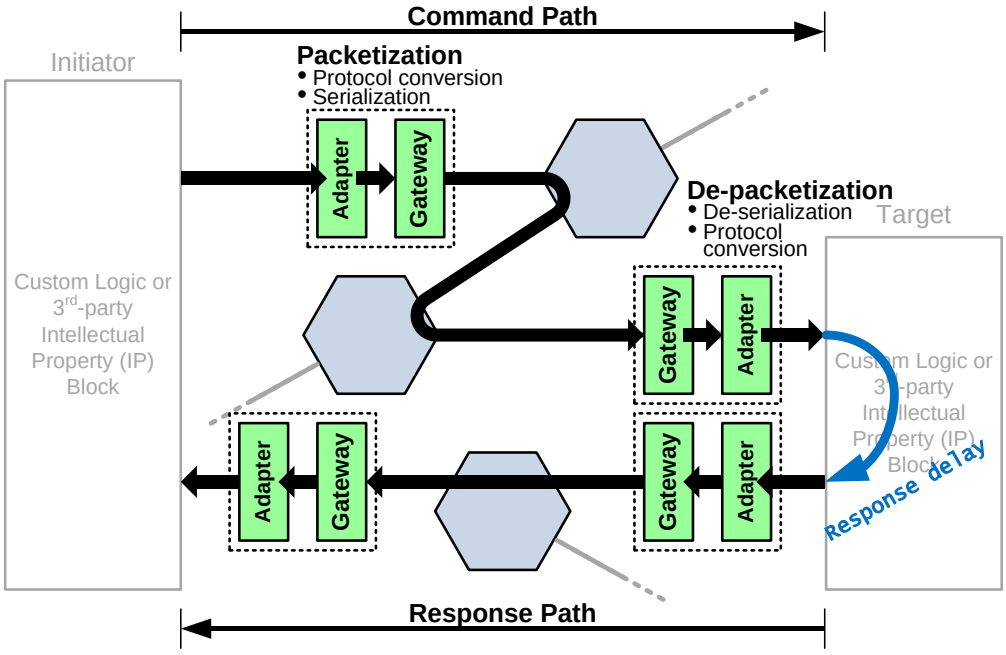
Roundtrip connection specified in CSL file

Command path

Response path

Violation

Figure 18: **An Example Roundtrip Connection Path**



Command Path network latency + Response Path network latency = **Network Roundtrip Latency**

Network Roundtrip Latency + Response delay = **System Roundtrip Latency**

Violations

The Violations section of the report file summarizes any violations encountered when checking the CSL file.

Figure 19 shows an example of a reported violation. In this particular case, look back to Figure 17 in the Roundtrip Connections section of the report file for the listed connection.

Figure 19: **Example Violations Listing from the Report File**

```
==========
Violations
==========
WARNING: System roundtrip latency requirement not met for
cpu_domain.CPU.CPU_initiator => memory_domain.SDRAM.SDRAM_target.
Slack is -19.503 ns
```

This particular violation is a warning. The network fails to meet the requested roundtrip latency, but does meet the other connectivity requirements. Figure 20 provides a snippet from the CSL file that caused the path violation. In the CSL file, the **write_reponse** delay for the SDRAM target is 25 ns. In the connection declaration, the target system roundtrip latency is 90 ns, which means that the available interconnect roundtrip latency must be 65 ns (90 ns – 25 ns). From the report file snippet shown in Figure 17, the actual interconnect roundtrip latency is 84.503 ns, leaving –19.503 ns of slack time. A negative slack time means that CHAINarchitect was not able to meet the requested latency goal. If a goal is missed, reconsider the overall system goals and explore other potential implementations.

Figure 20: **Snippet from CSL File that Caused Path Violation**

```
// SDRAM endpoint
    SDRAM {
     target SDRAM_target {
              . . .
         write_response = 25 ns;
         read_response = 50 ns;
         } // end SDRAM_target
    } // end SDRAM endpoint
  } // end memory_domain

              . . .

// Connections between CPU and SDRAM
CPU.CPU_initiator => SDRAM.SDRAM_target
                     (bandwidth=166 MBs, latency=90 ns);
```

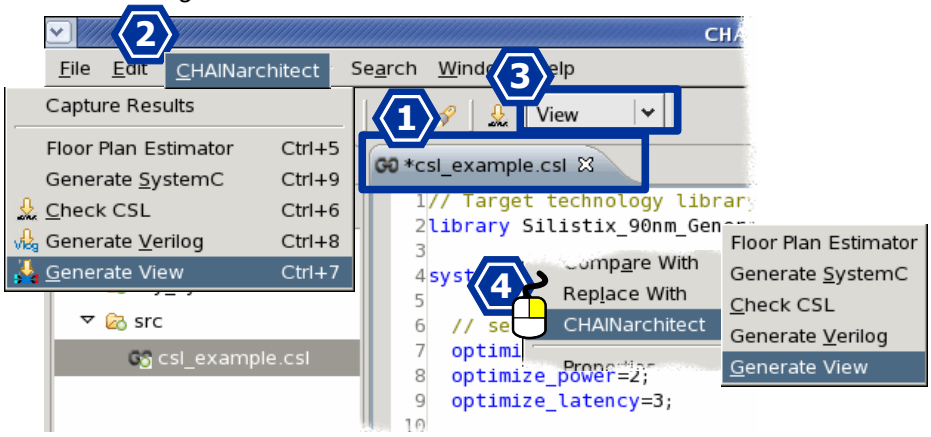
To see hints on how to improve the bandwidth or latency performance for a missed constraint, add the **--generate-hints** compiler option. See Setting CSL Compiler Options on page 39.

View a Generated Network

Generate View

1. Before you can view the network, you must first generate the view. To generate the view, you **must first click the CSL file tab** in the main window, as shown in Figure 21.

Figure 21: **Create Network View – Three Methods**



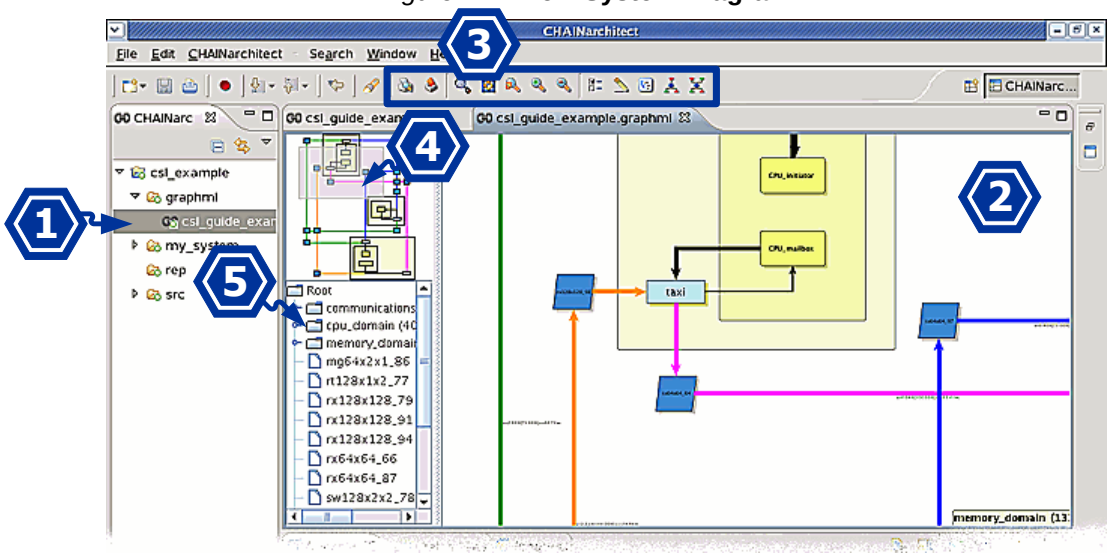
After selecting the CSL file, there are three possible ways to generate the network view, as listed below. The resulting network view appears in Figure 22.

2. **Method A:** From the main menu, click **CHAINarchitect → Generate View**.

3. **Method B:** Set the Go Button to **View**, then click the Go button, .

4. **Method C:** From within the CSL file, **right-click** and then select **CHAINarchitect → Generate View** from the pop-up menu.

Figure 22: **View System Diagram**



1. As shown in Figure 22, CHAINarchitect generates a diagram of the generated network. To view the diagram, expand the **graphml** folder under the current project. Double-click the underlying `*.graphml` file.

2. The system diagram appears in a separate tabbed window. The diagram shows the various domains, endpoints, targets, initiators, protocol adapters, and network connections as defined in the source CSL file.

3. With the system diagram open, the View toolbar appears, allowing you to zoom, rearrange, and print the diagram.

4. Regardless of the current zoom level, a small thumbnail diagram of the entire system appears to the left of the main display window. When zoomed in on the diagram in the main display, a small gray box surrounds corresponding region in the thumbnail diagram. To pan a zoomed image, click and drag the mouse in either the main display or on the thumbnail diagram.

5. A tree list displays the entire hierarchy of the network. All of the library components appear. Click to expand a domain.

View Toolbar

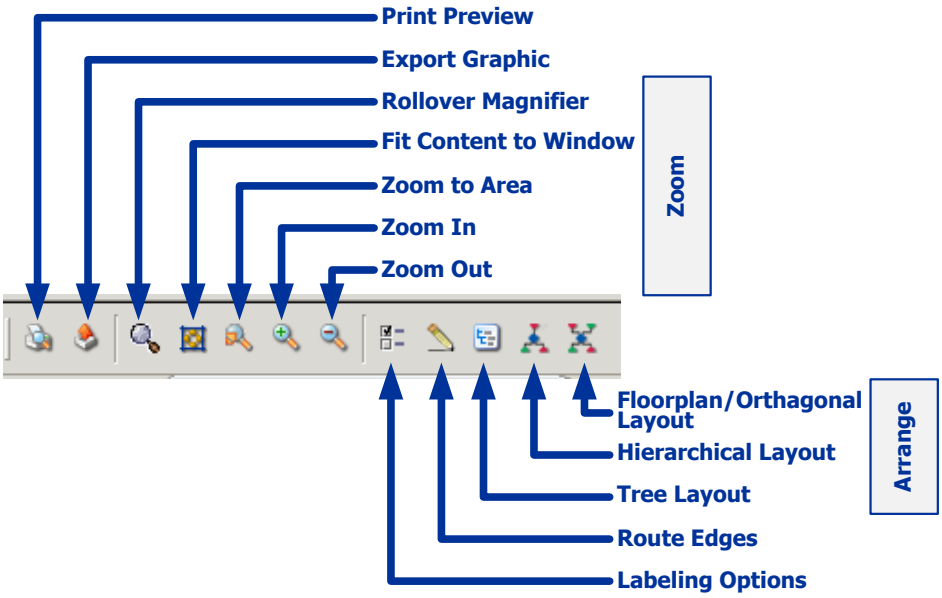
When a **`*.graphml`** file is open in a tabbed window, the View toolbar also appears below the top menu, as highlighted in Figure 22. Figure 23 labels each of the View toolbar items. The associated commands fit into one of three groups.

1. **Zoom Options:** Zoom the diagram in or out or zoom to a selected area. Optionally, fit the entire diagram in the display window or view selected features with the Rollover Magnifier.

2. **Arrangement Options:** Rearrange the diagram as a hierarchical diagram, a tree diagram, or as a "floorplan" view (although not as an integrated circuit floorplan).

3. **General Options:** Print the diagram or export the diagram to various industry-standard graphical formats.

Figure 23: **View Toolbar (appears only when *.graphml tab open)**



Zoom Options

The zoom options change the magnification of the diagram in the display window. Another related zoom feature is called the Rollover Magnifier, described on page 24.

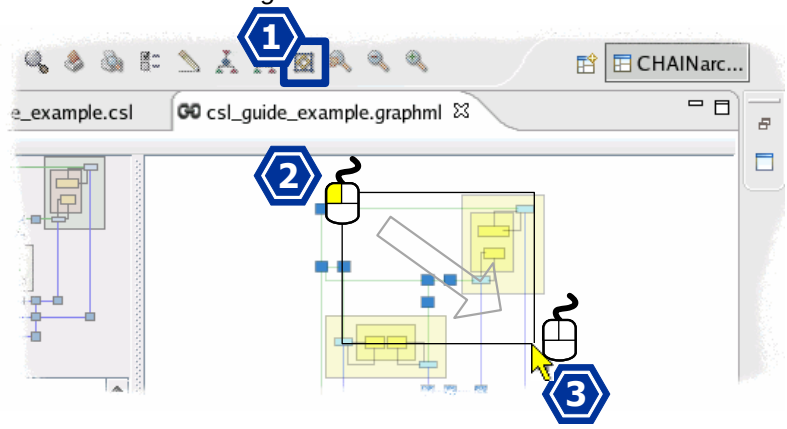
Zoom In, Zoom Out

The Zoom In and Zoom Out toolbar buttons magnify and shrink the diagram in the display window.

Zoom to Area

To zoom to a specific area, follow the steps listed below Figure 24.

Figure 24: **Zoom to Area**



1. As shown in Figure 24, click the Zoom to Area tool bar button.
2. Click and drag the mouse over the region of interest. A bounding box appears while dragging the mouse to highlight the area to be expanded.
3. Release the mouse button. The display zooms in to fit the selected area on the display.

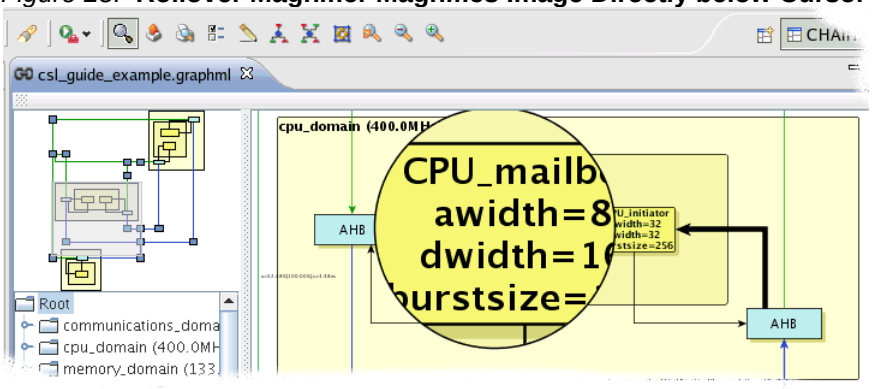
Fit Content to Window

Click the Fit Content to Window toolbar button to fit the entire diagram within the display window.

Rollover Magnifier

The rollover magnifier, demonstrated in Figure 25, magnifies the graphic immediately below the cursor. Click and toggle the magnifying lens icon in the View toolbar (see Figure 23) to enable and disable the rollover magnifier control.

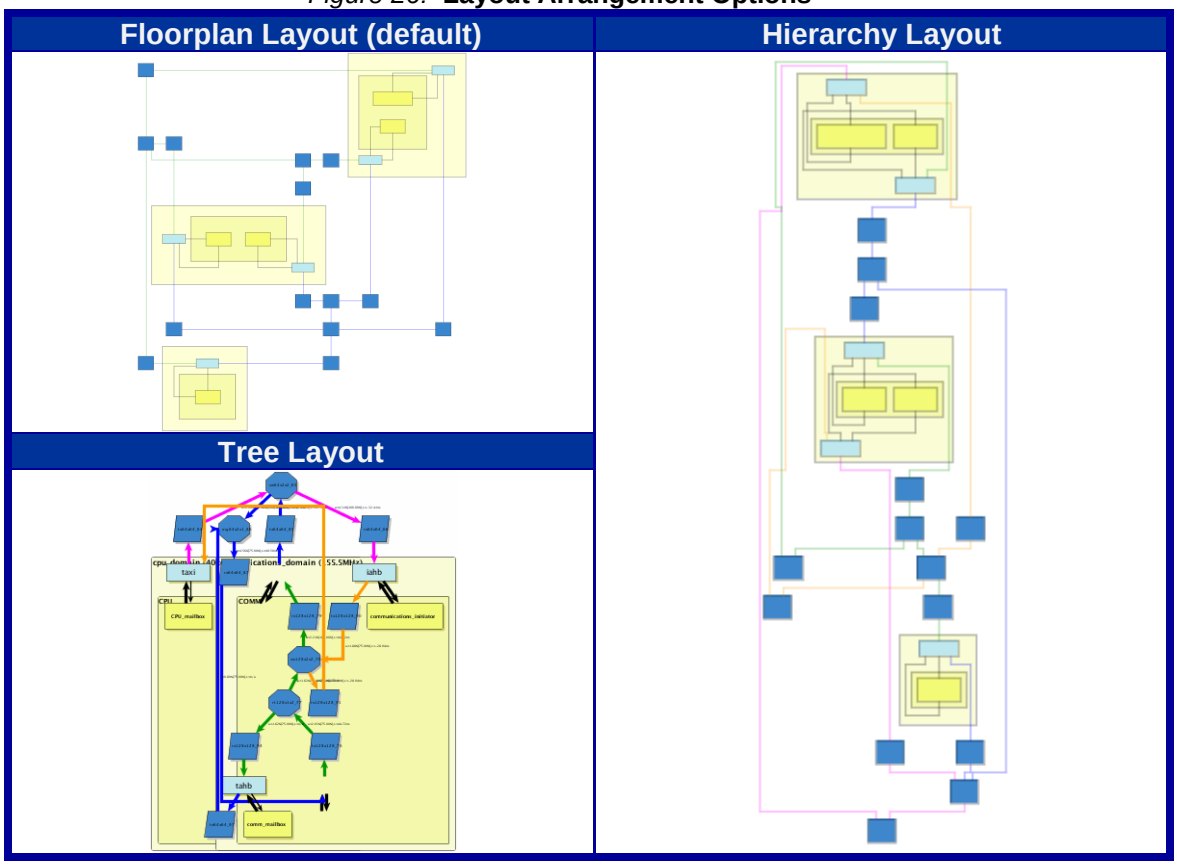
Figure 25: **Rollover Magnifier Magnifies Image Directly below Cursor**



Arrangement Options

The View display offers two different arrangements, as illustrated in Figure 26. The "floorplan" or "orthogonal" layout is the default view. Click the Hierarchy Layout toolbar button (see Figure 23) to change to the "hierarchy" layout, which appears much like a top-down organizational chart. Click the Tree Layout toolbar button (see Figure 23) to change to the "tree" layout.

Figure 26: **Layout Arrangement Options**

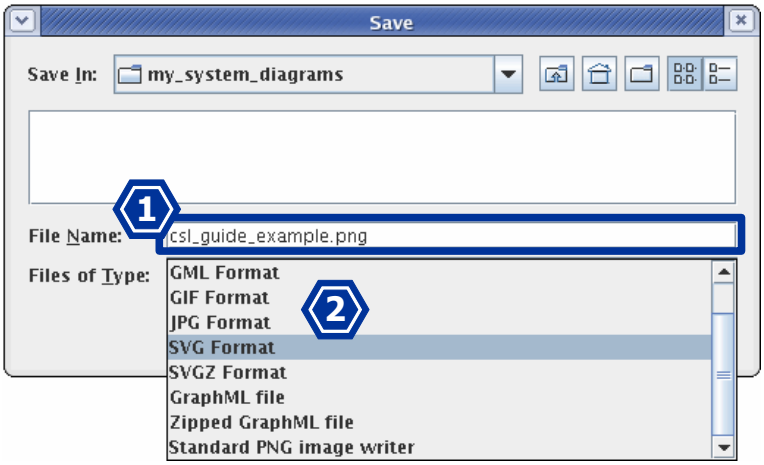


General Options

Export Graphic

To export the system diagram, click the Export Graphic toolbar button (see Figure 23).

Figure 27: **Export System Diagram Graphic**



1. As shown in Figure 27, specify a **File Name**.
2. Select the desired file format using the **Files of Type** drop list.
3. Click **Save** button.

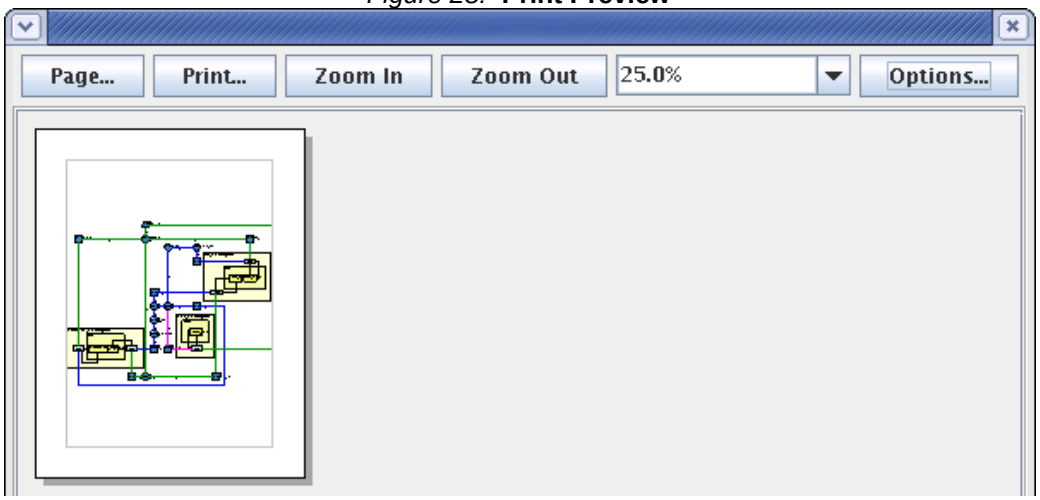
Change Export Graphic Settings

To change the graphic export preferences, click **Window → Preferences** from the CHAINarchitect menu, expand **CHAINarchitect Preferences** and select **Graph Export Preferences**.

Print Preview

To print the system diagram, click the Print Preview toolbar button (see Figure 23). This button opens the print preview panel, shown in Figure 28.

Figure 28: **Print Preview**



Using the **Options** button shown in Figure 28, choose whether to print the entire diagram (default) or the current display view.

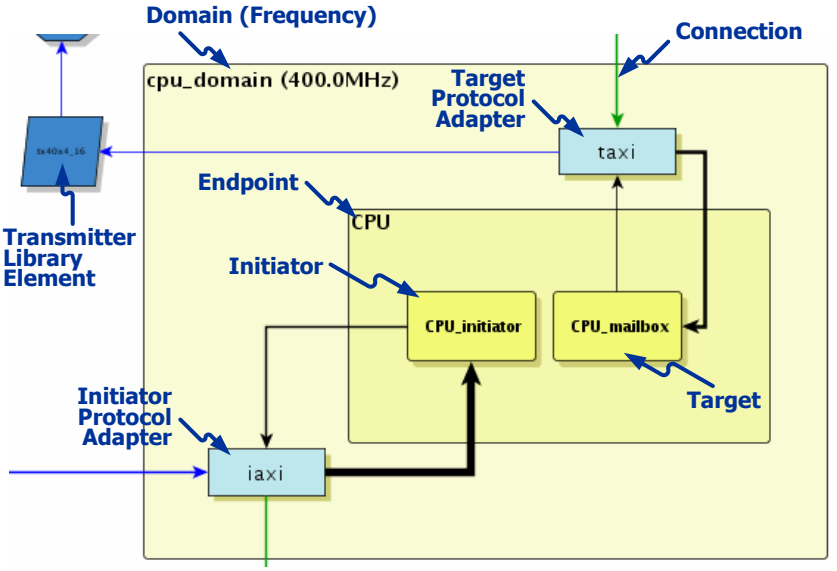
Interpreting the System Diagram View

The system diagram displays the network created by CHAINarchitect, generated from the CSL file specification. A few points of reference make it easier to interpret the resulting diagram.

Domain, Endpoint, Target, Initiator Connections

Each domain specified in the CSL source file is represented as a shaded area, as shown in Figure 29. Each endpoint in the CSL file is a shaded area within the domain. Finally, each initiator and target port is a shaded region within the endpoint region.

Figure 29: **System Diagram View of Domain and Endpoint Connections**



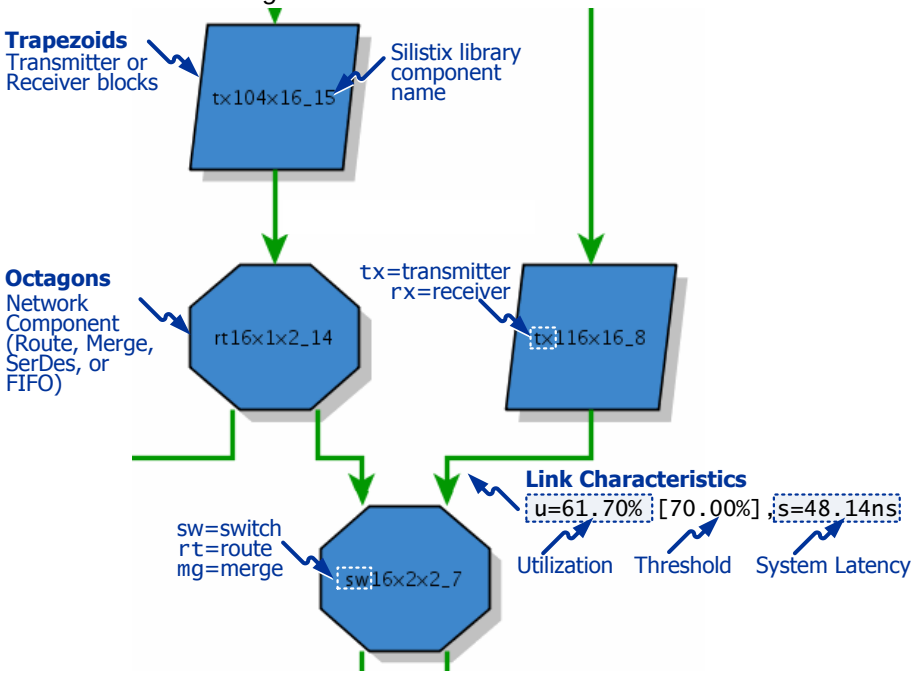
Interface protocol adapters, shaded in light blue, connect to the initiator and target ports. The first letter within the boxes represent the port type (`i`=initiator, `t`=target) while the remaining letters represent the protocol type (examples: `axi`, `ahb`, `apb`, `ocp`, etc.).

Network Connections and Characteristics

The various domain regions then connect to other domains via the Silistix network fabric, which is represented as blue trapezoids, blue octagons, and interconnection lines. Figure 30 shows a detailed snippet from a portion of the interconnect fabric.

The blue trapezoids in Figure 30 represent transmitter or receiver components within the Silistix network. One side of the trapezoid connects to an interface protocol adapter, as illustrated in the upper left corner of Figure 29. The other side of the trapezoid connects to other Silistix network library components. The text within each trapezoid is the component name and reference designator for the library component. The beginning of the component name starts with "`tx`" for transmitter or "`rx`" for receiver. These are the same names displayed in the Network Bill of Materials section of the report file and in the hierarchy tree browser shown in Figure 22.

Figure 30: **Silistix Network Connections**



The blue octagons represent various connections on the Silistix network fabric. These components are route connections, merge connections, switch connections, FIFO memories, or serializer/deserializer (SerDes) components. Again, the text within each octagon is the library name for the specific component.

Each line between components represents a network link. Each link has adjacent text that describes the characteristics of that connection segment. As shown in the lower right corner of Figure 30, each connection displays …

- its utilization of the available bandwidth on this specific link, represented as a percentage of total possible bandwidth (`u=61.70%`),
- the threshold for the link, as specified in the CSL source file but represented as a percentage, surrounded by square brackets (`[70.00%]`), and
- the system latency, in nanoseconds, through the preceding network component (`s=48.14ns`).

Connection Color-coding

The connections between network components are color-coded as shown in Table 2 to indicate traffic direction and whether there is a warning or error associated with a specific link or path.

Table 2: Connection Color Coding

Connection	Normal	Warning/Error
Initiator → Target	Blue	Purple
Initiator ← Target	Green	Orange

Generate First Placement Estimation

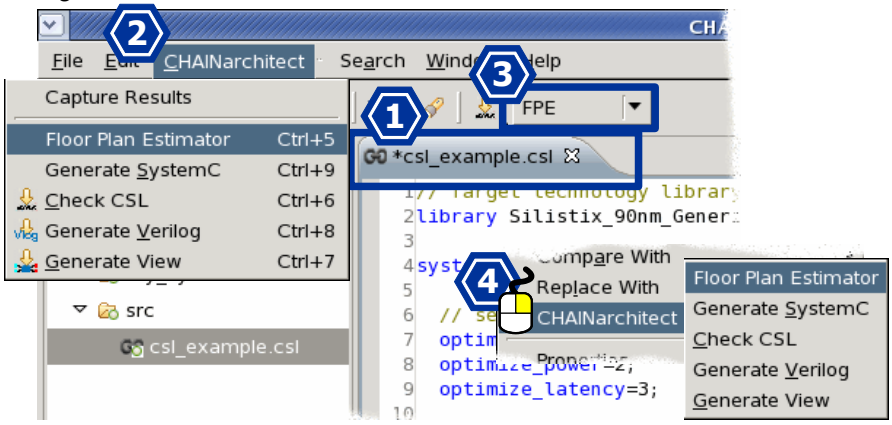
Based on specific statements in the CSL source file, CHAINarchitect generates a floor plan with initial placement for the system design. This floor plan allows CHAINarchitect to consider placement effects on the design and allows CHAINarchitect to automatically insert pipelatch components as appropriate to maintain network bandwidth over distance.

For more information on First Placement Estimator (FPE) commands available within the CSL language, see the following documents.

- ***Describing a System Using Connection Specification Language (CSL)***
- ***CSL Language Reference Manual***

1. To generate a First Placement Estimation, you **must first click the CSL file tab** in the main window, as shown in Figure 31.

Figure 31: **Generate First Placement Estimation – Three Methods**



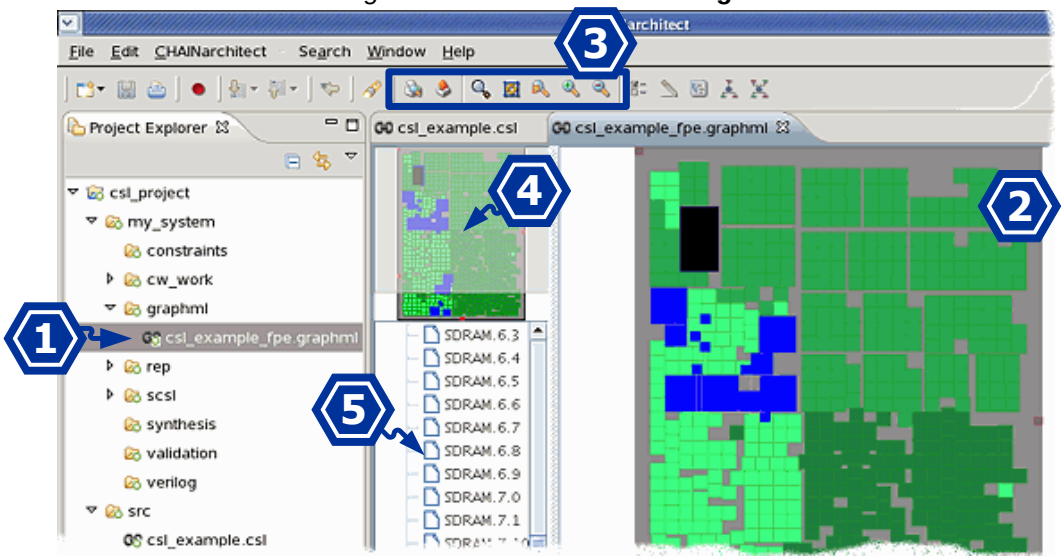
After selecting the CSL file, there are three possible ways to check the file, as listed below.

2. **Method A:** From the main menu, click **CHAINarchitect → Floor Plan Estimator**.

3. **Method B:** Set the Go Button to **FPE**, then click the Go button,  .

4. **Method C:** From within the CSL file, **right-click** and then select **CHAINarchitect → Floor Plan Estimator** from the pop-up menu.

Figure 32: **FPE Floor Plan Diagram**



1. As shown in Figure 35, CHAINarchitect generates a diagram of the generated floor plan. To view the diagram, expand the **graphml** folder under the current project. Double-click the underlying `*_fpe.graphml` file.

2. The floor plan diagram appears in a separate tabbed window. The color of each of the blocks has a specific meaning, described in Table 3. Similarly, the report file is updated based on the floor plan. See View and Interpret the CSL Compiler Report File on page 17.

3. With the floor plan diagram open, the FPE toolbar appears, allowing you to zoom and print the diagram. See FPE Toolbar.

4. Regardless of the current zoom level, a small thumbnail diagram of the entire floor plan appears to the left of the main display window. When zoomed in on the diagram in the main display, a small gray box surrounds corresponding region in the thumbnail diagram. To pan a zoomed image, click and drag the mouse in either the main display or on the thumbnail diagram.

5. A tree list displays the entire hierarchy of the network. All of the library components appear. Click to expand a domain.

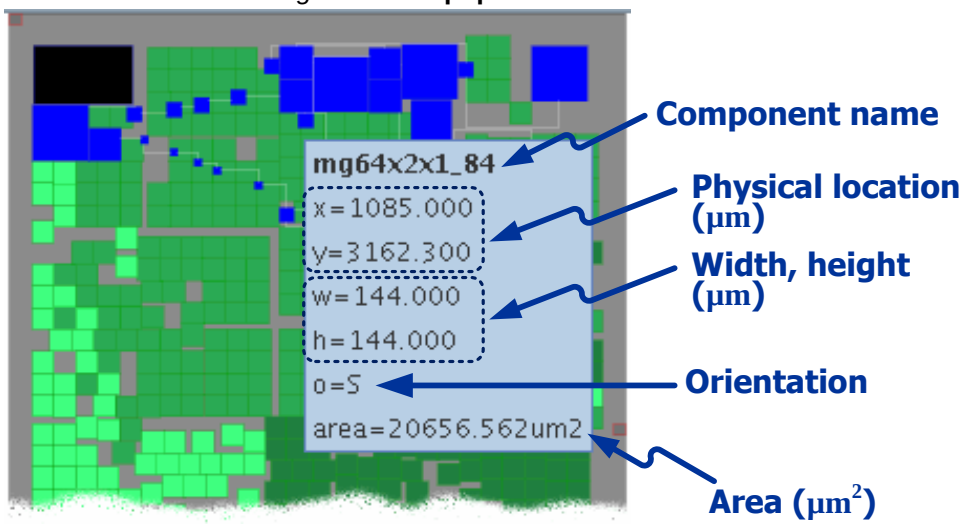
Table 3: **FPE Block Color Codes**

Color	Meaning
Blue	Silistix network component
Shades of Green	Soft IP block
Black	Hard IP block
Red Outline Around Block	Block with specific location

Popup Information Box

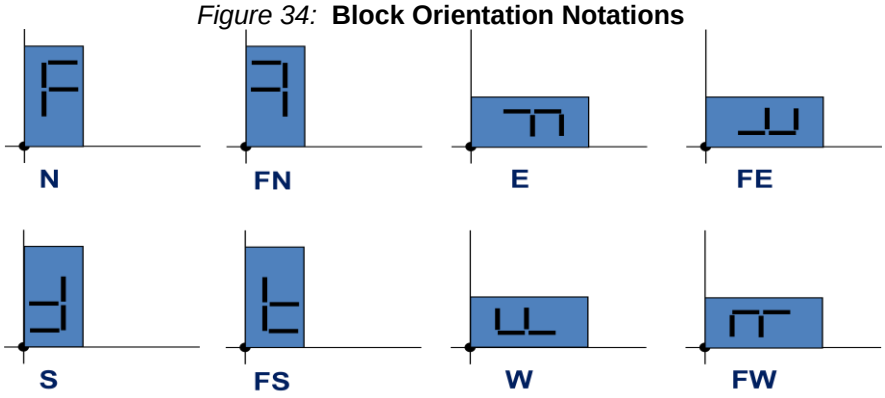
If the cursor lingers over the FPE floor plan drawing, CHAINarchitect displays a temporary popup information box, as shown in Figure 33. The box displays the name of the selected item, the physical location of its lower-left corner relative to the lower-left corner of the die, the width and height of selected item, and the area of the item.

Figure 33: **Popup Information Box**



For any hard IP blocks, such as Silistix network components, the information box also indicates the orientation of the block using a one- or two-character code. See Figure 34 for more information.

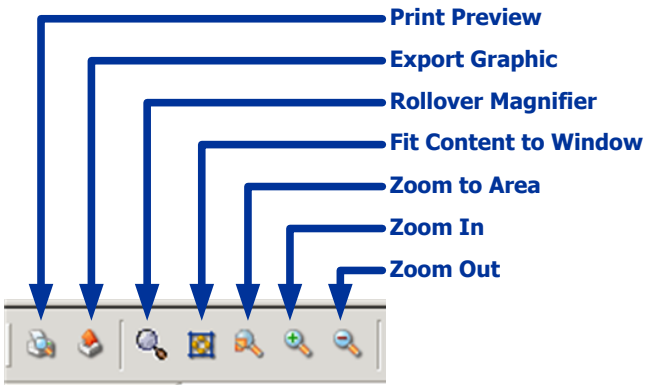
Figure 34: **Block Orientation Notations**



FPE Toolbar

When a `*_fpe.graphml` file is open in a tabbed window, the FPE toolbar appears as shown in Figure 35. The FPE zoom features behave as they do for the View function, as described in Zoom Options on page 24.

Figure 35: FPE Toolbar (only appears when a *_fpe.graphml file is open)



FPE Options

The First Placement Estimator (FPE) has a few additional controls as shown in Table 4. Set these options either on the CSL Compiler (CSLC) command line or via the CHAINarchitect CSL source-file properties, as described in Setting CSL Compiler Options on page 39.

Table 4: **First Placement Estimator Options**

Command Option	Description	Abbreviated Form
--perform-fpe[:<dbpath>]	Perform First Placement Estimation (FPE)	`-fpe`
--clean-fpe-database	Clean FPE Database	`-cfpe`
--fpe-center-gateways	Center Network Gateways within Soft IP Block	`-fpcgw`
--fpe-weights:<w1>_<w2>_<w3>_<w4>	Set FPE Weights for the various inter-block connections outlined in Table 5	`-wt:`
--fpe-sub-block-size:<size>	Sets the FPE Sub-block Size, in µm. Default is 30um.	`-sbs:`
--fpe-wrap-edges	Wrap Soft IP Block Edges	`-wse`
--no-pipelatch-insertion	No Pipelatch Insertion	`-nopl`
--generate-scsl[:filename]	Generate Structural CSL File	`-gf`
--generate-verilog	Generate DEF File	`-gv`

Perform First Placement Estimation (FPE)

The `--perform-fpe` option causes the CSL Compiler to run the First Placement Estimator.

Clean FPE Database

The `--clean-fpe-database` option clears the internal FPE database before running First Placement Estimation (FPE).

To reduce runtime, the First Placement Estimation software monitors the timestamp of the CSL source file relative to the most-recent FPE result. If the FPE result is newer, the Silistix software uses the placement information from the previous run and verifies that the resulting network meets or exceeds requirements.

Use the `--clean-fpe-database` option to reset the database before an FPE run, at the expense of a longer runtime.

Also use the `--clean-fpe-database` option when retrieving information back from the final physical design. In this instance, there is no need to perform placement, but only to insert any pipelatches required and check that the requirements are still met.

Center Network Gateways within Soft IP Block

The `--fpe-center-gateways` initially places the TX or RX gateways in center of any soft IP block or endpoint. By default, gateways are located at the edge. This is the initial placement only; the final placement may be different.

FPE Weights

The `--fpe-weights` option controls the net weights for various elements in the Silistix network and for soft IP blocks in the design. This option has four separate sub-controls as described in Table 5. Each sub-control supports a value between 1 and 1000. The higher the value, the more important is the connection type. Table 6 provides examples where each sub-control is heavily weighted.

```
--fpe-weights:<w1>_<w2>_<w3>_<w4>
```

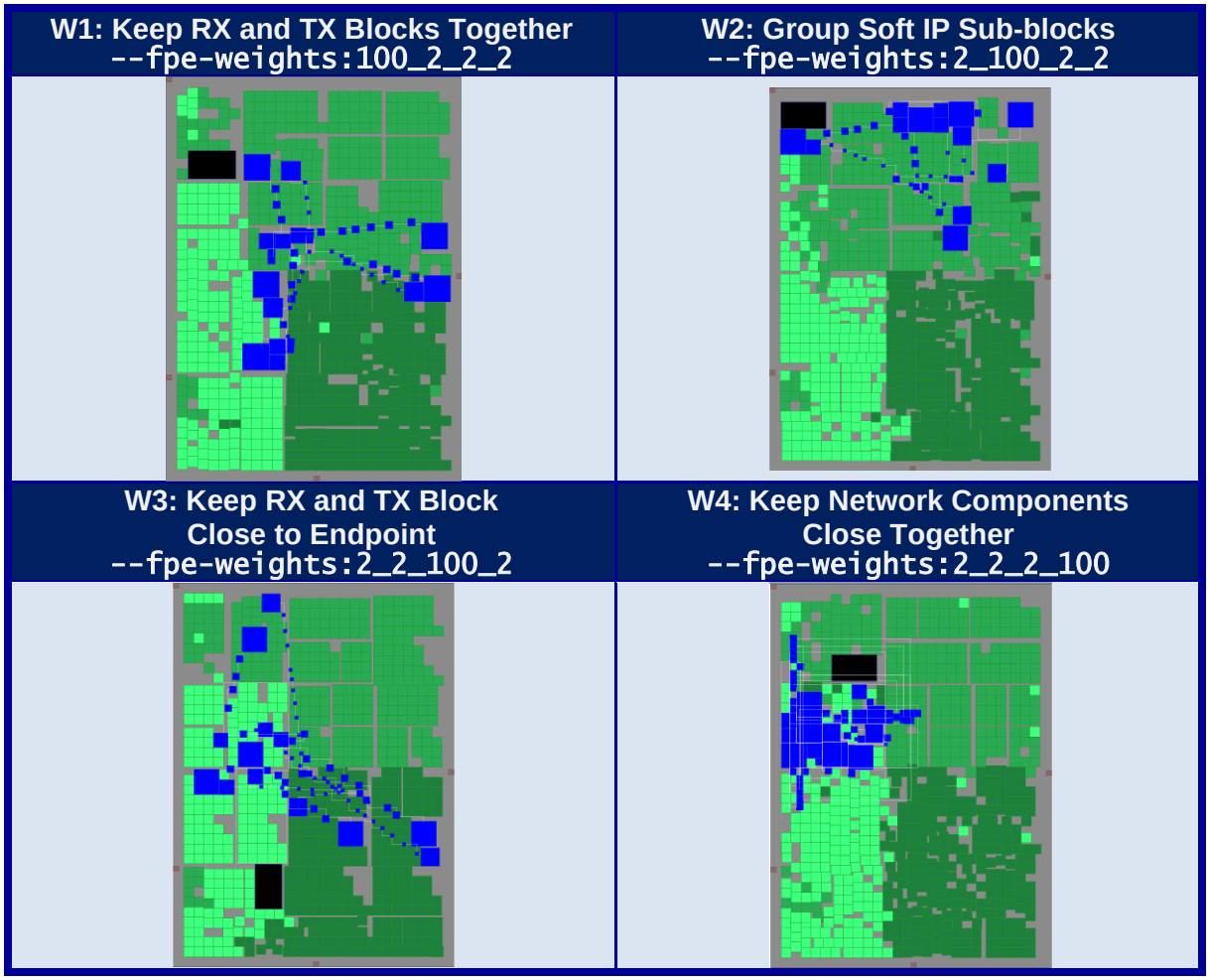
Example

```
--fpe-weights:2_50_2_100
```

Table 5: **FPE Weights**

Weight	Description
w1	**RX/TX Relationship:** Controls how closely the RX and TX blocks that service a specific adapter are placed to one another.
w2	**Soft IP Sub-block Relationship:** Controls how closely the mesh of soft logic blocks are place together as part of a soft IP block. A high value causes the sub-blocks within a soft IP to cluster together, squeezing out unrelated sub-blocks into other regions. Similarly, a high setting tends to pull the individual sub-blocks together rather than having them placed across the die.
w3	**RX/TX to Adapter Logic Relationship:** Controls how closely the RX and TX blocks within an adapter are placed to the adapter logic that connects to the endpoint logic.
w4	**Silistix Network Components Relationship:** Controls the placement of hard IP blocks within the Silistix network. A high value tends to pull the network into a smaller region and reduces the number of pipelatches. A low value allows the network elements more freedom, but at the expense of additional pipelatches. The effective weight also increases with the width of the network links.

Table 6: **FPE Weights Examples**



FPE Sub-block Size

FPE splits a soft IP block (`block-type = soft` or `soft_terminal`) into smaller sub-blocks of a specific size. If unspecified, the sub-blocks are 100 µm in size. Setting the sub-block to a smaller size may result in better overall placement estimations but also increases runtime. Table 7 provides examples comparing results and runtimes for different settings.

```
--fpe-sub-block-size:<size>
```

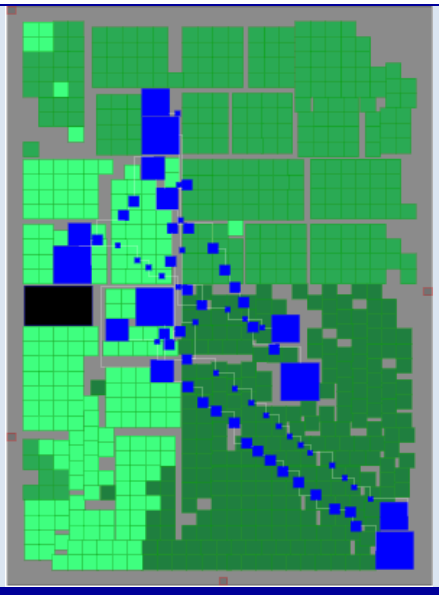
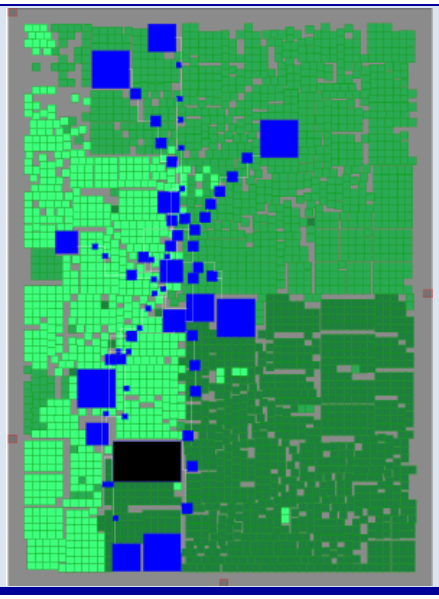
Where:

<size> is the sub-block size specified in µm.

Example

```
--fpe-sub-block-size:50
```

Table 7: **Sub-block Size Examples**

Option	None (default)	`--fpe-sub-block-size:50`
Sub-block Size	100 µm	50 µm
Run Time	1X	~5X
Example Results		

Wrap Soft IP Block Edges

The `--fpe-wrap-edges` option only affects soft IP blocks and it only has a significant effect on larger soft IP endpoints with complex placement constraints.

This option causes the CSL Compiler to generate connection wires between the sub-blocks on the horizontal and vertical edges of the mesh of sub-blocks. This provides the placer with more freedom to rotate, or even to turn the sub-block mesh inside out for a more realistic placement.

No Pipelatch Insertion

Use the `--no-pipelatch-insertion` option to skip automatic pipelatch insertion.

Generate Structural CSL File

The `-generate-scscl` option produces a structural CSL file that includes all modules, all parameter settings for all modules, and includes all the network components, including an inserted pipelatch components.

Generate DEF File

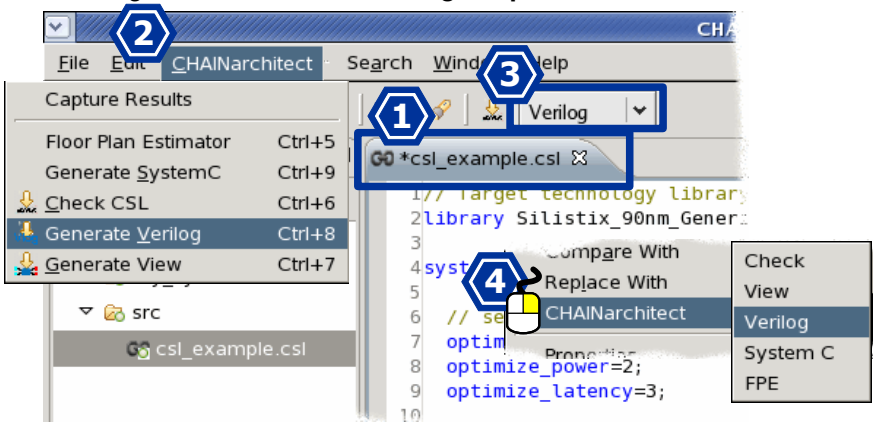
To generate a DEF placement file, include the `--generate-verilog` compiler option.

The resulting `*.def` file is saved under the **constraints** subdirectory.

Generate Verilog Structural Netlist and Validation Models

1. To generate the Verilog output files, you **must first click the CSL file tab** in the main window, as shown in Figure 36.

Figure 36: **Generate Verilog Output – Three Methods**



After selecting the CSL file, there are three possible ways to check the file, as listed below.

2. **Method A:** From the main menu, click **CHAINarchitect → Generate Verilog**.

3. **Method B:** Set the Go Button to **Verilog**, then click the Go button,  .

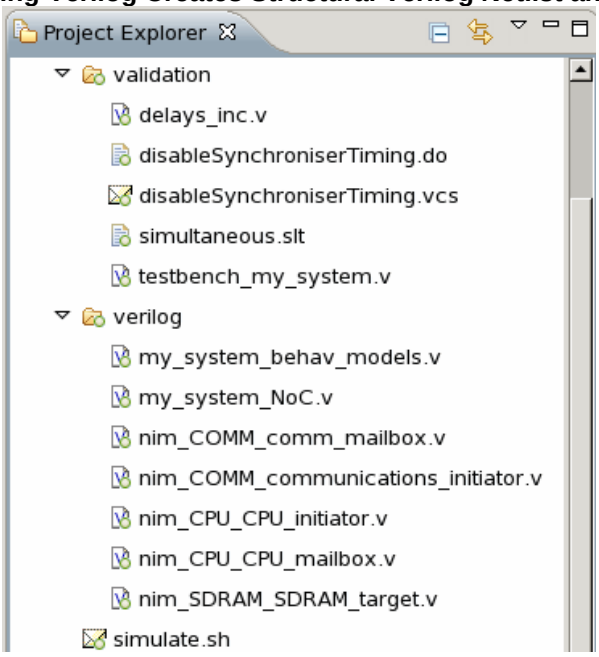
4. **Method C:** From within the CSL file, **right-click** and then select **CHAINarchitect → Generate Verilog** from the pop-up menu.

When complete, this operation by default generates synthesizable structural Verilog netlists, Verilog behavioral models, and Verilog simulation files, along with various stimulus and script files. See Setting CSL Compiler Options on page 39 to change or add to these standard options.

As shown in Figure 37, this operation creates a new subdirectory under the project directory, using the name from the `system` statement in the CSL file. In the example design, the system is called `my_system`.

The behavioral and synthesizable, structural Verilog netlist files are saved in the **verilog** subdirectory under the **<project>/<system>** directory. Similarly, the Verilog simulation files appear in the **validation** subdirectory. Double-click on a file to view it as a tabbed window in the main window.

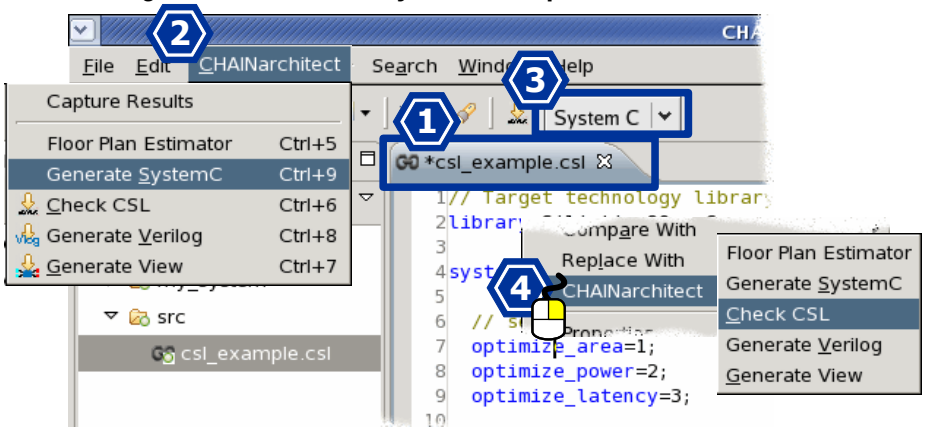
Figure 37: **Generating Verilog Creates Structural Verilog Netlist and Simulation Files**



Generate SystemC Validation Models

1. To generate the SystemC output files, you **must first click the CSL file tab** in the main window, as shown in Figure 38.

Figure 38: **Generate SystemC Output – Three Methods**



After selecting the CSL file, there are three possible ways to check the file, as listed below.

2. **Method A:** From the main menu, click **CHAINarchitect → Generate System C**.

3. **Method B:** Set the Go Button to **System C**, then click the Go button,  .

4. **Method C:** From within the CSL file, **right-click** and then select **CHAINarchitect → Generate SystemC** from the pop-up menu.

When complete, this operation generates SystemC simulation files, along with various stimulus and script files.

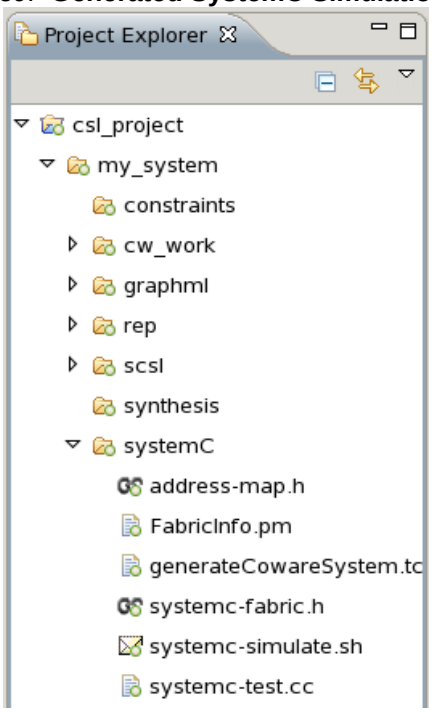
As shown in Figure 39, this operation creates a new subdirectory under the project directory, using the name from the `system` statement in the CSL file. In the example design, the system is called `my_system`.

The SystemC files are saved in the /**systemC** subdirectory under the /**my_system** subdirectory. Double-click on a file to view it as a tabbed window in the main window.

If using the CoWare SystemC simulator, set the CSL Compiler option **`-sc:CoWare`**, as shown in Figure 41 and described in the Setting CSL Compiler Options section, page 39.

To successfully generate the CoWare simulation files, the CoWare software must be in your current setup environment.

Figure 39: **Generated SystemC Simulation Files**

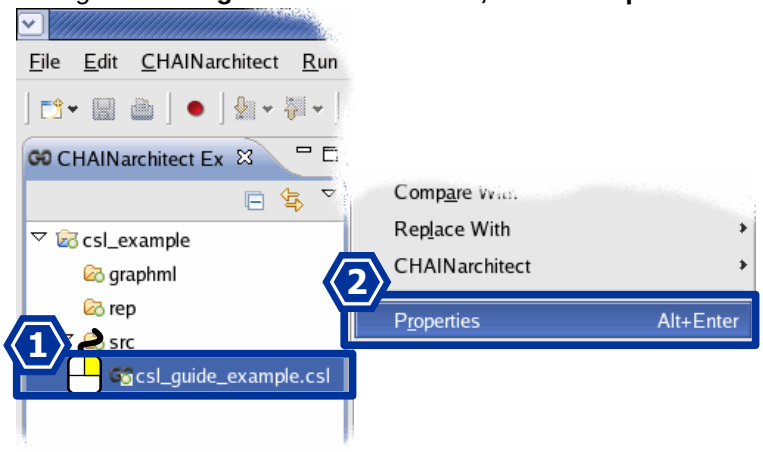


Setting CSL Compiler Options

The Silistix CSL Compiler underlies all CHAINarchitect operations. To set CSL Compiler software options for the specific CSL file, follow the steps outlined after Figure 40.

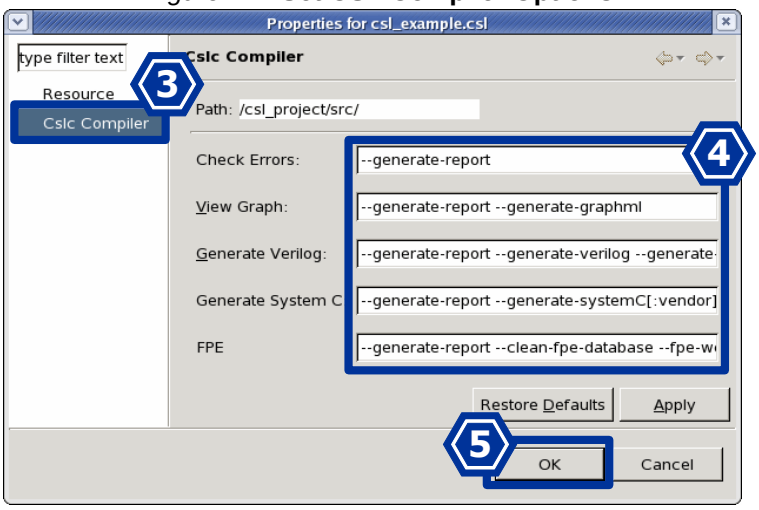
To set the CSL Compiler options for all new projects, select **Window → Preferences** from the CHAINarchitect menu. Choose **CHAINarchitect Preferences** and enter the command-line settings in the **Compiler options** text box. However, any general preferences are overruled by CSL Compiler options set for the specific CSL file.

Figure 40: **Right-click on CSL File, Select Properties**



1. As shown in Figure 40, expand the project tree to reveal the CSL file. Right-click on the CSL file name.
2. Select **Properties** from the resulting pop-up menu.

Figure 41: **Set CSL Compiler Options**



3. As shown in Figure 41, click **Cslc Compiler** to reveal the available option settings.

4. Modify the available options as desired. See the available options shown in Table 8. Optionally, invoke the CSL Compiler help screen by typing `cslc -help` in a console window.

5. When finished, click **OK** to save the new option settings.

Once these options are changed, they are automatically saved with the associated file. If a file compiles differently than expected using the default settings, check to see if there are other option settings saved with the file.

Table 8: **CSL Compiler Options**

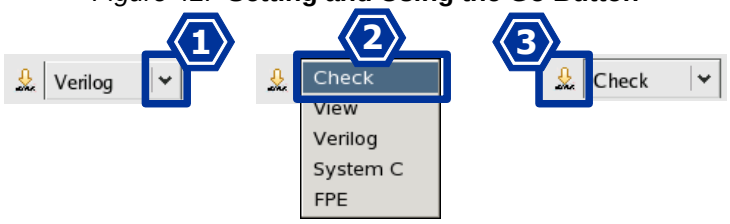
Option	Description	Abbreviated Form
Generators		
--generate-hints	Generate bandwidth hints.	`-gh`
--generate-scsl[:<fname>]	Generate structural csl file.	`-gf`
--generate-graphml[:<fname>]	Generate graphml file.	`-og`
--generate-report[:<fname>]	Generate report file.	`-or`
--generate-graph	Generate graphical representation of topology.	`-dg`
--generate-olb	Generate compiled object library.	`-olb`
--generate-systemC[:vendor]	Generate behavioral SystemC model.	`-sc`
--generate-timed-systemC	Generate timed SystemC model.	`-tm`
--generate-verilog	Generate Verilog.	`-gv`
--generate-behavioral	Generate behavioral Verilog models.	`-bm`
Synthesis		
--synopsys-scripts[:tool]	Generate Synopsys synthesis scripts.	`-synopsys`
--magma-scripts[:tool]	Generate Magma synthesis scripts.	`-magma`
--cadence-scripts[:tool]	Use Cadence for synthesis.	`-cadence`
Optimization Control		
--optimize-latency:<n>	Set latency optimization priority.	`-ol`
--optimize-area:<n>	Set area optimization priority.	`-oa`
--optimize-power:<n>	Set power optimization priority.	`-op`
--disable-tree-balancing	Disable tree balancing optimization.	`-db`
--disable-utilization	Disable utilization optimization.	`-du`
--utilization-threshold:<n>	Set the utilization threshold.	`-ut`
--frequency-ratio-optimization	Enable frequency ratio optimization.	`-efro`
--no-frequency-ratio-matching	Disable frequency ratio matching optimization.	`-dfrm`
--no-edge-serdes-optimization	Disable edge serdes optimization.	`-deso`
NPV		
--generate-npv-traffic[:<dir>]	Generate NPV traffic files.	`-npv`
--npv-threshold:<tolerance>	NPV bandwidth tolerance.	`-npvt`
--npv-iterations:<n>	NPV minimum traffic iteration count for initiators. Default is 10.	`-npvi`
--fast-npv-traffic	Generate NPV traffic for faster (less accurate) simulation	

Option	Description	Abbreviated Form
--distribute-npv-traffic	Distribute NPV traffic	
--randomize-npv-traffic	Randomly stagger NPV traffic	
FPE		
--perform-fpe[:<dbpath>]	Perform FPE.	**-fpe**
--clean-fpe-database[:<dbpath>]	Clean FPE database.	**-cfpe**
--fpe-seed:<seed>	Set FPE seed value	**-fpseed**
--fpe-center-gateways	Center gateways within SIP blocks.	**-fpcgw**
--fpe-weights:<w1>_<w2>_<w3>_<w4>	Set net weights for FPE. w1=Tx/Rx w2=SIP w3=GW w4=Async.	**-wt**
--fpe-sub-block-size:<sz>	Sub-block size for FPE graph. Default is 100um.	**-sbs**
--fpe-wrap-edges	Wrap edges of soft IP blocks.	**-wse**
--no-pipelatch-insertion	Insert no pipelatches.	**-nopl**
--load-def[:<filename>]	Load DEF file.	**-def**
Metrics		
--capture-metrics[:<fname>]	Write metrics to capture file.	
Misc		
--report-gate-area	Report area in kgates.	**-ga**
--defined-hard-macros-only	Use only hard macros define in LEF file.	**-dhm**
--no-warnings	Suppresses display of warnings.	**-nw**
--verbose	Display verbose progress	**-v**
--define:<macro>[=<value>]	Define preprocessing macro and optional value.	**-d**
--help	Display command line help.	**-h**
--quiet	Quiet mode.	**-q**

Go Button

The Go button, when set up, provides a quick, single-click method to generate file output. The steps following Figure 42 describe how to set up and use the go button.

Figure 42: **Setting and Using the Go Button**



1. As shown in Figure 42, click the down arrow to reveal the drop list of possible Go button functions.
2. Select the desired function for the Go button from the drop list. The selection remains set until changed.
3. Whenever you wish to execute the desired function, click the Go button.

Revision History

Revision	Date	Description/Revisions
1.2.2	12-DEC-2008	Minor corrections.
1.2.1	31-OCT-2008	Minor update for NPV.
1.2	5-AUG-2008	Added Generate First Placement Estimation section. Updated for CHAINworks 2.1 release.
1.1	22-MAY-2008	Various updates to match latest release.
1.0	21-DEC-2007	Initial release.

Feedback

Feedback on this Silistix document and all Silistix products is highly encouraged. If you have a comment, correction, or suggestion to improve this document, please send us an E-mail. Please include complete details including page numbers, section titles, or figure or table numbers where appropriate. Thank you in advance for helping us to improve our products and services.

feedback@silistix.com

Disclaimers

Software Licensing Statements

This section contains copyright statements and licenses that apply to various open source libraries that may be used by components of Silistix CHAINworks.

ANTLR

ATK - Accessibility Toolkit

ATK - Accessibility Toolkit
Copyright 2001 Sun Microsystems Inc.

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Library General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Library General Public License for more details.

You should have received a copy of the GNU Library General Public License along with this library; if not, write to the Free Software Foundation, Inc., 59 Temple Place - Suite 330, Boston, MA 02111-1307, USA.

Cairo

Copyright 1998, 2000, 2004 Keith Packard

Permission to use, copy, modify, distribute, and sell this software and its documentation for any purpose is hereby granted without fee, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation, and that the name of Keith Packard not be used in advertising or publicity pertaining to distribution of the software without specific, written prior permission. Keith Packard makes no representations about the suitability of this software for any purpose. It is provided "as is" without express or implied warranty.

KEITH PACKARD DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE, INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS, IN NO EVENT SHALL KEITH PACKARD BE LIABLE FOR ANY SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

Copyright 2004,2005 Red Hat, Inc.
Copyright 2005 Trolltech AS

Permission to use, copy, modify, distribute, and sell this software and its documentation for any purpose is hereby granted without fee, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation, and that the name of Red Hat not be used in advertising or publicity pertaining to distribution of the software without specific, written prior permission. Red Hat makes no representations about the suitability of this software for any purpose. It is provided "as is" without express or implied warranty.

THE COPYRIGHT HOLDERS DISCLAIM ALL WARRANTIES WITH REGARD TO THIS SOFTWARE, INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS, IN NO EVENT SHALL THE COPYRIGHT HOLDERS BE LIABLE FOR ANY SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

Copyright 2003 Carl Worth

Permission to use, copy, modify, distribute, and sell this software and its documentation for any purpose is hereby granted without fee, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation, and that the name of Carl Worth not be used in advertising or publicity pertaining to distribution of the software without specific, written prior permission. Carl Worth makes no representations about the suitability of this software for any purpose. It is provided "as is" without express or implied warranty.

CARL WORTH DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE, INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS, IN NO EVENT SHALL CARL WORTH BE LIABLE FOR ANY SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

Copyright 1987, 1998 The Open Group

Permission to use, copy, modify, distribute, and sell this software and its documentation for any purpose is hereby granted without fee, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation.

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE OPEN GROUP BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Except as contained in this notice, the name of The Open Group shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization from The Open Group.

Copyright 1987 by Digital Equipment Corporation, Maynard, Massachusetts.
All Rights Reserved

Permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in· supporting documentation, and that the name of Digital not be used in advertising or publicity pertaining to distribution of the software without specific, written prior permission.

DIGITAL DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE, INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS, IN NO EVENT SHALL DIGITAL BE LIABLE FOR ANY SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

FontConfig

FreeType 2

- The FreeType License, found in the file `FTL.TXT', which is similarto the original BSD license *with* an advertising clause that forces you to explicitly cite the FreeType project in your product's documentation. All details are in the license file. This license is suited to products which don't use the GNU General Public License.
- The GNU General Public License version 2, found in `GPL.TXT' (any later version can be used also), for programs which already use the GPL. Note that the FTL is incompatible with the GPL due to its advertisement clause.

The contributed PCF driver comes with a license similar to that of the X Window System. It is compatible to the above two licenses (see file src/pcf/readme).

The FreeType Project LICENSE
2002-Apr-11
Copyright 1996-2002 by David Turner, Robert Wilhelm, and Werner Lemberg

Introduction

The FreeType Project is distributed in several archive packages; some of them may contain, in addition to the FreeType font engine, various tools and contributions which rely on, or relate to, the FreeType Project.

This license applies to all files found in such packages, and which do not fall under their own explicit license. The license affects thus the FreeType font engine, the test programs, documentation and makefiles, at the very least.

This license was inspired by the BSD, Artistic, and IJG (Independent JPEG Group) licenses, which all encourage inclusion and use of free software in commercial and freeware products alike. As a consequence, its main points are that:

- We don't promise that this software works. However, we will be interested in any kind of bug reports. (`as is' distribution)
- You can use this software for whatever you want, in parts or full form, without having to pay us. (`royalty-free' usage)
- You may not pretend that you wrote this software. If you use it, or only parts of it, in a program, you must acknowledge somewhere in your documentation that you have used the FreeType code. (`credits')

We specifically permit and encourage the inclusion of thissoftware, with or without modifications, in commercial products.

We disclaim all warranties covering The FreeType Project and assume no liability related to The FreeType Project.

Finally, many people asked us for a preferred form for a credit/disclaimer to use in compliance with this license. We thus encourage you to use the following text:

```
"""
 Portions of this software are copyright © 1996-2002 The FreeType
 Project (www.freetype.org).  All rights reserved.
"""
```

Legal Terms

0. Definitions

Throughout this license, the terms `package', `FreeType Project', and `FreeType archive' refer to the set of files originally distributed by the authors (David Turner, Robert Wilhelm, and Werner Lemberg) as the `FreeType Project', be they named as alpha, beta or final release.

`You' refers to the licensee, or person using the project, where `using' is a generic term including compiling the project's source code as well as linking it to form a `program' or `executable'.

This program is referred to as `a program using the FreeType engine'.

This license applies to all files distributed in the original FreeType Project, including all source code, binaries and documentation, unless otherwise stated in the file in its original, unmodified form as distributed in the original archive. If you are unsure whether or not a particular file is covered by this license, you must contact us to verify this.

The FreeType Project is copyright (C) 1996-2000 by David Turner, Robert Wilhelm, and Werner Lemberg. All rights reserved except as specified below.

1. No Warranty

THE FREETYPE PROJECT IS PROVIDED `AS IS' WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. IN NO EVENT WILL ANY OF THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY DAMAGES CAUSED BY THE USE OR THE INABILITY TO USE, OF THE FREETYPE PROJECT.

2. Redistribution

This license grants a worldwide, royalty-free, perpetual and irrevocable right and license to use, execute, perform, compile, display, copy, create derivative works of, distribute and sublicense the FreeType Project (in both source and object code forms) and derivative works thereof for any purpose; and to authorize others to exercise some or all of the rights granted herein, subject to the following conditions:

- Redistribution of source code must retain this license file (`FTL.TXT') unaltered; any additions, deletions or changes to the original files must be clearly indicated in accompanying documentation. The copyright notices of the unaltered, original files must be preserved in all copies of source files.
- Redistribution in binary form must provide a disclaimer that states that the software is based in part of the work of the FreeType Team, in the distribution documentation. We also encourage you to put an URL to the FreeType web page in your documentation, though this isn't mandatory.

These conditions apply to any software derived from or based on the FreeType Project, not just the unmodified files. If you use our work, you must acknowledge us. However, no fee need be paid to us.

3. Advertising

Neither the FreeType authors and contributors nor you shall use the name of the other for commercial, advertising, or promotional purposes without specific prior written permission.

We suggest, but do not require, that you use one or more of the following phrases to refer to this software in your documentation or advertising materials: `FreeType Project', `FreeType Engine', `FreeType library', or `FreeType Distribution'.

As you have not signed this license, you are not required to accept it. However, as the FreeType Project is copyrighted material, only this license, or another one contracted with the authors, grants you the right to use, distribute, and modify it. Therefore, by using, distributing, or modifying the FreeType Project, you indicate that you understand and accept all the terms of this license.

4. Contacts

There are two mailing lists related to FreeType:

- freetype@freetype.org

Discusses general use and applications of FreeType, as well as future and wanted additions to the library and distribution. If you are looking for support, start in this list if you haven't found anything to help you in the documentation.

- devel@freetype.org

Discusses bugs, as well as engine internals, design issues, specific licenses, porting, etc.

- http://www.freetype.org

Holds the current FreeType web page, which will allow you to download our latest development version and read online documentation.

You can also contact us individually at:

- David Turner david.turner@freetype.org
- Robert Wilhelm robert.wilhelm@freetype.org
- Werner Lemberg werner.lemberg@freetype.org

GC

Copyright 1988, 1989 Hans-J. Boehm, Alan J. Demers
Copyright (c) 1996-1999 by Silicon Graphics. All rights reserved.
Copyright (c) 1991-1995 by Xerox Corporation. All rights reserved.
Copyright 1999-2004 by Hewlett-Packard Company. All rights reserved.

THIS MATERIAL IS PROVIDED AS IS, WITH ABSOLUTELY NO WARRANTY EXPRESSEDOR IMPLIED. ANY USE IS AT YOUR OWN RISK.

Permission is hereby granted to use or copy this program for any purpose, provided the above notices are retained on all copies.

Permission to modify the code and to distribute modified code is granted, provided the above notices are retained, and a notice that the code was modified is included with the above copyright notice.

Copyright (c) 1996-1997 Silicon Graphics Computer Systems, Inc.
Copyright (c) 2002 Hewlett-Packard Company

Permission to use, copy, modify, distribute and sell this software and its documentation for any purpose is hereby granted without fee, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation. Silicon Graphics makes no representations about the suitability of this software for any purpose. It is provided "as is" without express or implied warranty.

GLib

Copyright (C) 1995-1997, 2002 Peter Mattis, Spencer Kimball and Josh MacDonald
Copyright (C) 2003 Sebastian Wilhelmi
Copyright (C) 1998, 2000-2003 Tim Janik
Copyright (C) 1998-2005 Red Hat Inc
Copyright 2001 Hans Breuer
Copyright 2003, 2005 Matthias Clasen
Copyright (C) 1999-2003 Free Software Foundation, Inc.
Copyright (C) 2004 Anders Carlsson <andersca@gnome.org>
Copyright 2004 Tor Lillqvist
Copyright (C) 2003 Sebastian Wilhelmi
Copyright (C) 1999, 2000 Tom Tromey
Copyright (C) 2003 Noah Levitt
Copyright (C) 2005 Imendio AB

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public License along with this library; if not, write to the Free Software Foundation, Inc., 59 Temple Place - Suite 330, Boston, MA 02111-1307, USA.

GTK

org.graphdrawing.graphml.util.Base64

MetaPlacer

The Microelectronics Advanced Research Corporation (MARCO), the Gigascale Silicon Research Center (GSRC), the Defense Advanced Research Projects Agency, the University of Michigan and the University of California ("PROVIDERS") MAKE NO WARRANTIES, whether express or implied, including warranties of merchantability or fitness for a particular purpose or noninfringement, with respect to this software and supporting documentation.

Providers have NO obligation to provide ANY support, assistance, installation, training or other services, updates, enhancements or modifications related to this software and supporting documentation.

Providers shall NOT be liable for ANY costs of procurement of substitutes, loss of profits, interruption of business, or any other direct, indirect, special, consequential or incidental damages arising from the use of this software and its documentation, whether or not Providers have been advised of the possibility of such damages.

MPFR

Copyright 1991, 1993-1997, 1999-2004 Free Software Foundation, Inc.

The MPFR Library is free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 2.1 of the License, or (at youroption) any later version.

The MPFR Library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public License along with the MPFR Library; see the file COPYING.LIB. If not, write to the Free Software Foundation, Inc., 59 Temple Place - Suite 330, Boston, MA 02111-1307, USA.

Pango

Copyright 1999-2005 Red Hat Software
Copyright (C) 1996-2001 Sun Microsystems
Copyright 1996-2000 D. Turner, R.Wilhelm, and W. Lemberg
Copyright (C) 2002 Sven Neumann
Copyright (C) 2002-2005 Changwoo Ryu
Copyright (c) 2001 by Sun Microsystems, Inc.
Copyright (C) 1998-2003 IBM Corporation. All Rights Reserved.
Copyright (C) 2004 Emil Soleyman-Zomalan
Copyright (C) 2004 Theppitak Karoonboonyanan
Copyright (C) 2002 Software and Language Engineering Laboratory, NECTEC
Copyright (C) 1999,2000 Dov Grobgeld
Copyright (C) 2001,2002 Behdad Esfahbod
Copyright (C) 2000 Tor Lillqvist
Copyright (C) 2001 Alexander Larsson
Copyright (C) 2000 SuSE Linux Ltd

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.» See the GNU Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public License along with this library; if not, write to the Free Software Foundation, Inc., 59 Temple Place - Suite 330, Boston, MA 02111-1307, USA.

PNG

This copy of the libpng notices is provided for your convenience. In case of any discrepancy between this copy and the notices in the file png.h that is included in the libpng distribution, the latter shall prevail.

COPYRIGHT NOTICE, DISCLAIMER, and LICENSE:

If you modify libpng you may insert additional notices immediately following this sentence.

libpng version 1.2.6, December 3, 2004, is Copyright (c) 2004 Glenn Randers-Pehrson, and is distributed according to the same disclaimer and license as libpng-1.2.5 with the following individual added to the list of Contributing Authors

- Cosmin Truta

libpng versions 1.0.7, July 1, 2000, through 1.2.5 - October 3, 2002, are Copyright (c) 2000-2002 Glenn Randers-Pehrson, and are distributed according to the same disclaimer and license as libpng-1.0.6 with the following individuals added to the list of Contributing Authors

- Simon-Pierre Cadieux
- Eric S. Raymond
- Gilles Vollant

and with the following additions to the disclaimer:

There is no warranty against interference with your enjoyment of the library or against infringement. There is no warranty that our efforts or the library will fulfill any of your particular purposes or needs. This library is provided with all faults, and the entire risk of satisfactory quality, performance, accuracy, and effort is with the user.

libpng versions 0.97, January 1998, through 1.0.6, March 20, 2000, are Copyright (c) 1998, 1999 Glenn Randers-Pehrson, and are distributed according to the same disclaimer and license as libpng-0.96, with the following individuals added to the list of Contributing Authors:

- Tom Lane
- Glenn Randers-Pehrson
- Willem van Schaik

libpng versions 0.89, June 1996, through 0.96, May 1997, are Copyright (c) 1996, 1997 Andreas Dilger/ Distributed according to the same disclaimer and license as libpng-0.88, with the following individuals added to the list of Contributing Authors:

- John Bowler
- Kevin Bracey
- Sam Bushell
- Magnus Holmgren
- Greg Roelofs
- Tom Tanner

libpng versions 0.5, May 1995, through 0.88, January 1996, are Copyright (c) 1995, 1996 Guy Eric Schalnat, Group 42, Inc. For the purposes of this copyright and license, "Contributing Authors" is defined as the following set of individuals:

- Andreas Dilger
- Dave Martindale
- Guy Eric Schalnat
- Paul Schmidt
- Tim Wegner

The PNG Reference Library is supplied "AS IS". The Contributing Authors and Group 42, Inc. disclaim all warranties, expressed or implied, including, without limitation, the warranties of merchantability and of fitness for any purpose. The Contributing Authors and Group 42, Inc. assume no liability for direct, indirect, incidental, special, exemplary, or consequential damages, which may result from the use of the PNG Reference Library, even if advised of the possibility of such damage.

Permission is hereby granted to use, copy, modify, and distribute this source code, or portions hereof, for any purpose, without fee, subject to the following restrictions:

1. The origin of this source code must not be misrepresented.

2. Altered versions must be plainly marked as such and must not be misrepresented as being the original source.

3. This Copyright notice may not be removed or altered from any source or altered source distribution.

The Contributing Authors and Group 42, Inc. specifically permit, without fee, and encourage the use of this source code as a component to supporting the PNG file format in commercial products. If you use this source code in a product, acknowledgment is not required but would be appreciated.

Libpng is OSI Certified Open Source Software. OSI Certified Open Source is a certification mark of the Open Source Initiative.

Glenn Randers-Pehrson
glennrp at users.sourceforge.net
December 3, 2004

zlib

Copyright (C) 1995-2003 Mark Adler

This software is provided 'as-is', without any express or implied warranty. In no event will the author be held liable for any damages arising from the use of this software.

Permission is granted to anyone to use this software for any purpose, including commercial applications, and to alter it and redistribute it freely, subject to the following restrictions:

1. The origin of this software must not be misrepresented; you must not claim that you wrote the original software. If you use this software in a product, an acknowledgment in the product documentation would be appreciated but is not required.
2. Altered source versions must be plainly marked as such, and must not be misrepresented as being the original software.
3. This notice may not be removed or altered from any source distribution.