Allegro Design Editor Tutorial

Product Version 15.5.1
December 2005

Cadence Design Systems, Inc., 555 River Oaks Parkway, San Jose, CA 95134, USA

Contents

Preface

This chapter discusses the following:

Purpose of This Tutorial

This tutorial provides lessons, sample design files, and multimedia demonstrations to help new users learn how to use Allegro Design Editor.

The tutorial contains the following modules:

- Module 1: Working with Projects on page 21
- Module 2: Working with Components and Connectivity on page 55
- Module 3: Working with Associated Components on page 111
- Module 4: Working with Properties on page 145
- Module 5: Working with Electrical Constraints on page 175
- Module 6: Creating a Hierarchical Design on page 215
- Module 7: Generating Reports on page 249

Audience

This tutorial is designed to get you quickly started on Allegro Design Editor. This tutorial assumes that you are familiar with the development and design of electronic circuits at the system or board level. This tutorial also assumes a working knowledge of the following Cadence tools:

- Allegro Design Entry HDL
- Allegro Constraint Manager
- Allegro PCB Editor
- Allegro SigXplorer

Software Requirements

To perform all the exercises in this tutorial, you need the following Cadence tools:

- Allegro Design Editor

- Allegro Design Entry HDL
- Allegro Constraint Manager
- Allegro PCB Editor
- Allegro SigXplorer

How to Use This Tutorial

The training is offered in three learning modes:

- Written lessons provide detailed procedures for performing basic operations.
- Multimedia presentations demonstrate the written procedures.
- Sample design files offer a starting point for practicing with the tools.

Depending on your personal learning style, you can use this tutorial in different ways:

- You might begin by reading through the written tutorial lessons. After completing each lesson, watch the multimedia demonstration to enhance your understanding of the procedures. Then, work through the procedures yourself using the sample design files with Design Editor.

  Note: The exercises in this tutorial are built on a sample design and are progressive in nature. Follow the lessons in the sequence used in this tutorial. Skipping lessons might unsynchronize the design with the screenshots given in the tutorial.

- Another approach you can take is to watch the multimedia demonstrations first to gain a general understanding of how to work with the tools. Then, as you experiment with the sample files using Design Editor, you can refer to the written lessons to refresh your memory about procedures you saw in the demonstrations.

The written lessons, demonstrations, and sample designs all work together to reinforce your learning experience. Use them in a way you find most conducive to learning.

Understanding the Sample Design Files

You can load the sample design files into Design Editor and begin working with them immediately. The lessons and multimedia demonstrations use these same design files to illustrate the procedures. You can work with the design files as you progress through the lessons.

Before you can use the sample design files, install the design files and set the required environment variables as described below.

Important

All the steps described in this tutorial assume that the sample design files are installed in the `c:\designs` folder. If you install the sample design files at any other location, the file locations described in the tutorial steps and the file locations displayed in the screenshots in the tutorial will change accordingly.

Installing the Design Files

To use the design files, copy them to your system using the following instructions:

On Microsoft Windows

1. Create a directory in which you want to install the design files. For this tutorial, we will refer to this directory as <*your_work_area*>.
2. Copy the `ade_tut_db.zip` file located in the directory:

   <*install_dir*>/doc/ade_tut/tutorial_examples

   to the <*your_work_area*> directory.
3. Unzip the `ade_tut_db.zip` file.

UNIX and Linux

1. For this tutorial, we will refer to this directory as <*your_work_area*>.

2. Copy the `ade_tut_db.t.Z` file located in the directory:

   `<*your_inst_dir*>\doc\ade_tut\tutorial_examples`

   to the <*your_work_area*> directory.

3. Uncompress and untar the `ade_tut_db.t.Z` file.

Note: For more information about the tutorial files, see **List of Sample Design Files** on page 282.

Setting Environment Variables

After installing the design files, you need to set the following environment variables on your system:

Environment Variable	Description
TUTORIAL_LIB	This environment variable should point to: <*your_work_area*>/reference/ref_lib
TDD_START_PROJ_LOCATION	This environment variable should point to: <*your_work_area*> The TDD_START_PROJ_LOCATION environment variable specifies the default location in which all new projects will be created. For example, if you specify `c:\designs` as the value for the `TDD_START_PROJ_LOCATION` environment variable, and create a new project using the New Project Wizard in Design Editor, the project will be created in the `c:\designs` folder, unless you specify a different location for the project.

You must also set the `PADPATH` and `PSMPATH` Allegro environment variables by doing the following:

1. Type the following command in the Windows or UNIX command prompt:

   `enved`

   The User Preferences dialog box appears.

2. In the *Categories* list, click the *Design_paths* tab.

3. Click the *Value* button next to the `padpath` variable.

   The padpath items dialog box appears.

4. Add the following path:

   `<your_work_area>/reference/pcb/symbols`

5. Click *OK*.

6. Click the *Value* button next to the `psmpath` variable.

   The psmpath items dialog box appears.

7. Add the following path:

   `<your_work_area>/reference/pcb/symbols`

8. Click *OK*.

9. Click *OK* to close the User Preferences dialog box.

Understanding Multimedia Demonstrations

The multimedia demonstrations that accompany the tutorial lessons offer visual ways to grasp concepts and techniques that are described in the procedures. The demonstrations support and illustrate the procedures.

You can launch multimedia demonstrations in three ways:

- Click the hyperlink in the Multimedia Demonstration section in the beginning of a module or preceding the procedure for each lesson.
- Go to the section List of Multimedia Demonstrations on page 283 and click the hyperlink for the demo that you want to run.
- Open the Allegro Design Editor Start page and click the *DEMOS* tab. Then click the links in the *DEMOS* tab to view the demos.

Getting the Flash Player

To view the multimedia demonstrations, you need to install the appropriate Macromedia Flash Player on your system. Macromedia Flash Player is free and is available for different operating systems at:

http://www.macromedia.com/shockwave/download/alternates/

You can download the appropriate Macromedia Flash Player for Windows, Solaris, HP-UX, and Linux operating systems.

Important

The multimedia demonstrations included in this tutorial do not run on IBM AIX systems. However, the tutorials are designed to be used without viewing the demonstrations. You can follow procedures in the lessons while you perform the tasks using the sample designs.

Related Information

You can obtain additional information about how to use Design Editor from the following online manuals. You can access these manuals from the Cadence Program Group and from the Help menus of individual tools.

- *Allegro Design Editor User Guide*

Note: At the end of each lesson, you will find references to related sections of the *Allegro Design Editor User Guide*.

Syntax Conventions

This list describes the syntax conventions used in this tutorial.

`literal`	Key words that you must enter literally. These keywords represent commands (functions, routines) or option names.
`Courier font`	Command line examples.
UI	Menus, labels, fields, or tabs in the user interface.
`variable`	Arguments for which you must substitute a value.

Module 1: Working with Projects

Prerequisite

Install the sample design files and set the environment variables as described in Understanding the Sample Design Files on page 14.

Lessons

This module consists of the following lessons:

Multimedia Demonstration

Click the link below to view a Flash-based multimedia demonstration of this module.

Creating a Project in Allegro Design Editor

Completion Time

1 hour for written lessons

8 minutes for multimedia demonstrations

Prerequisite

Follow the instructions in Understanding the Sample Design Files on page 14 to install the sample design files and set the required environment variables.

Overview

The first task you perform in designing a PCB is to create a design project. A design project is the encapsulation of paths to libraries, part tables, tool settings, project-level settings, global settings and other related settings for designing a PCB to required specifications.

A design project consists of the following:

- Reference libraries
- Project libraries
- `cds.lib` file
- Project file (`.cpm` file)

The following sections provide more details about these components.

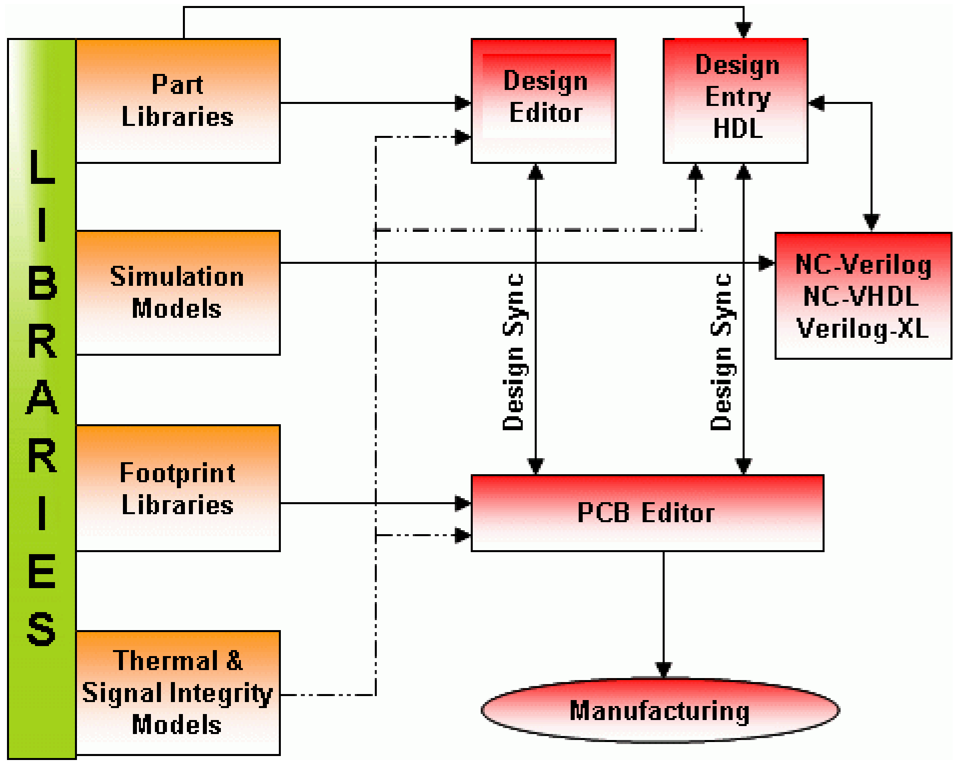
What are Libraries?

You begin the design process by creating a logic design using Allegro Design Editor or Allegro Design Entry HDL and then a board-level design that translates the logic design into a manufacturable entity. To accomplish this process, tools need a *software* representation of the various parts to be used in the design. The representations of these parts are organized into libraries.

The different tools used in the various stages of the design flow, need different views or information about the same part. Some of these views are schematic, footprint, and simulation.

These views are organized into several libraries. For example, footprints of various parts are consolidated into a single layout library.

The library organization for Cadence PCB design tools is as follows:



- Part libraries

  These libraries contain views for design entry or schematic creation. The information contained in these views includes logical symbols (graphical representations of the part), pinouts, and packaging information.

- Footprint libraries

  These libraries contain the footprints that correspond to the physical parts specified in part libraries. These libraries are required at the layout stage of the design flow.

- Simulation Libraries

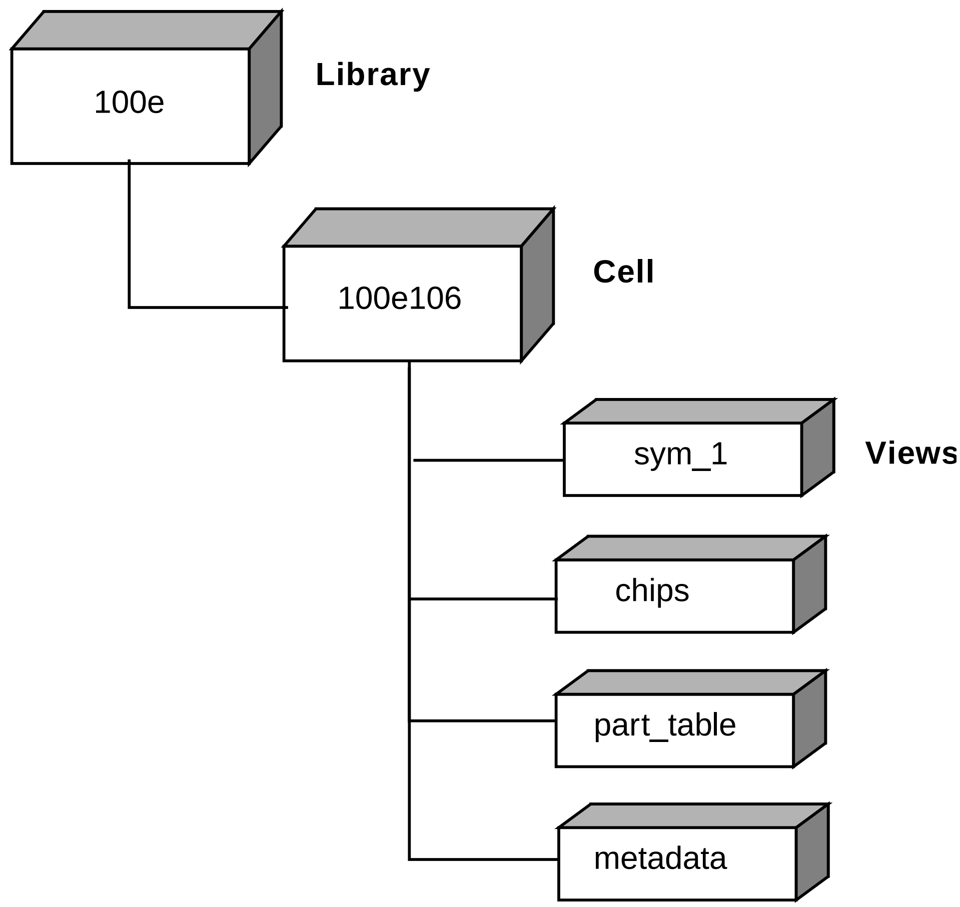
  These libraries model the behavior of the part in the Verilog or VHDL Hardware Description Languages. These libraries are required during the design verification phase.

Reference Libraries

Cadence supplies a set of `reference` libraries that contain views of parts belonging to several logic families. The `lsttl` library is an example of a reference library. Reference libraries are usually stored in an area to which you do not have write permissions and are managed by a librarian. The Cadence reference libraries are located at `<`*`your_inst_dir`*`>/share/library`

The following figure illustrates the structure of a reference library.

Figure 1-1 Library Organization: Reference Libraries



The following table describes each view within a part in a reference library.

View Name	Description
sym_1	Contains the schematic symbol. When you add a part to a design, a reference to the symbol view is placed in the design only (not the actual part).
chips	Maps the logical part to the physical package (pinouts). You can also use this view to assign other physical properties (for example, input and output loading characteristics, and voltages).
part_table	Lets you add custom part properties to fit your company needs. For example, you can add a company part number, part description or any in-house or vendor information you require.
metadata	Contains version information for the part.

Project Libraries

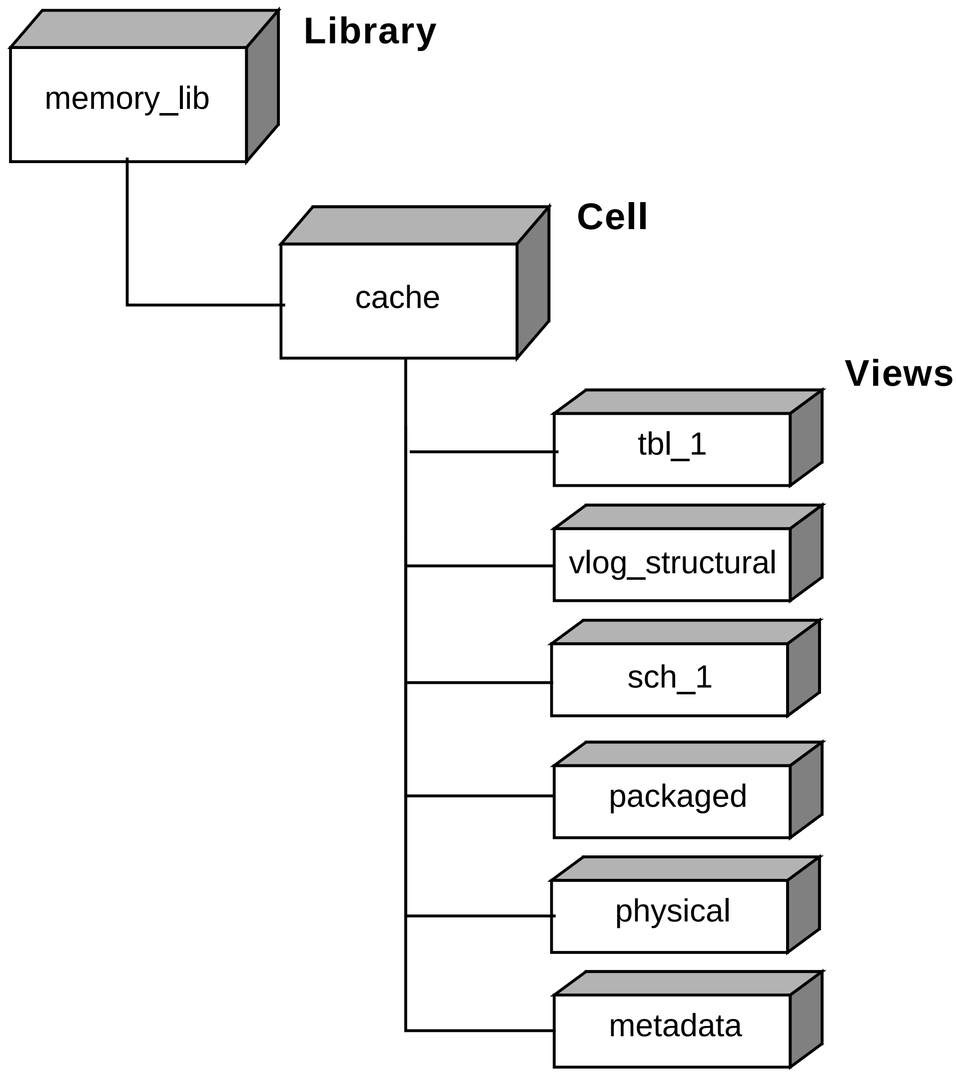
Project libraries (also known as local or design libraries) are used by designers at the project level. Project libraries contain parts customized for your project and the logical design you capture in Design Editor.

The libraries in which the designs you capture in Design Editor are stored are known as working libraries. By default, the directory for the working library for a project has the name `<`*`projectname`*`>_lib`. For example, if you create a project named `memory.cpm`, the working library for the project will be named `memory_lib`.

Each design is stored in a subdirectory within a working library. This subdirectory is known as a cell. A cell can represent the entire design or just a portion of the design (or hierarchy). Each cell or design contains subdirectories that represent different design phases (known as cell views).

The following figure illustrates the structure of a project library.

Figure 1-2 Library Organization: Project Libraries



The following table describes each view in a design library.

View	Description
tbl_1	Contains the table based design.
sch_1	Contains the schematics.
vlog_structural	Contains the Verilog description of the design.
packaged	Contains the results of packaging.
physical	Contains the PCB layout.

View	Description
metadata	Contains version information for the design.

What is a cds.lib File?

Design Editor is a by-reference design editor. This means that Design Editor references all parts in the design from various libraries that reside at the reference or project area.

The `cds.lib` file is the library definition file that defines all the libraries used in your design and maps them to their physical locations. The contents of a typical `cds.lib` file is given below:

```
DEFINE 54alsttl ../../library/54alsttl
DEFINE 54fact ../../library/54fact
DEFINE tutorial_lib worklib
DEFINE local_lib local_lib
INCLUDE $CHDL_LIB_INST_DIR/share/cdssetup/cds.lib
```

For more information on the `cds.lib` file, see the *Design Entry HDL Libraries Reference*.

What is a Project File

When you create a new project, Design Editor creates a project file called `<`*`projectname`*`>.cpm` in the project directory. The `<`*`projectname`*`>.cpm` file includes the following setup information for your project:

- The name of the top-level design and the library in which it is located.
- The list of project libraries.
- The location of the temporary directory where tools generate intermediate data
- Setup directives for Design Editor, Design Entry HDL, PCB Editor, and any other tool launched from the project.

Lesson 1-1: Starting Design Editor

Overview

In this lesson, you will learn to start Design Editor.

Procedure

To start Design Editor on Microsoft Windows, do one of the following:

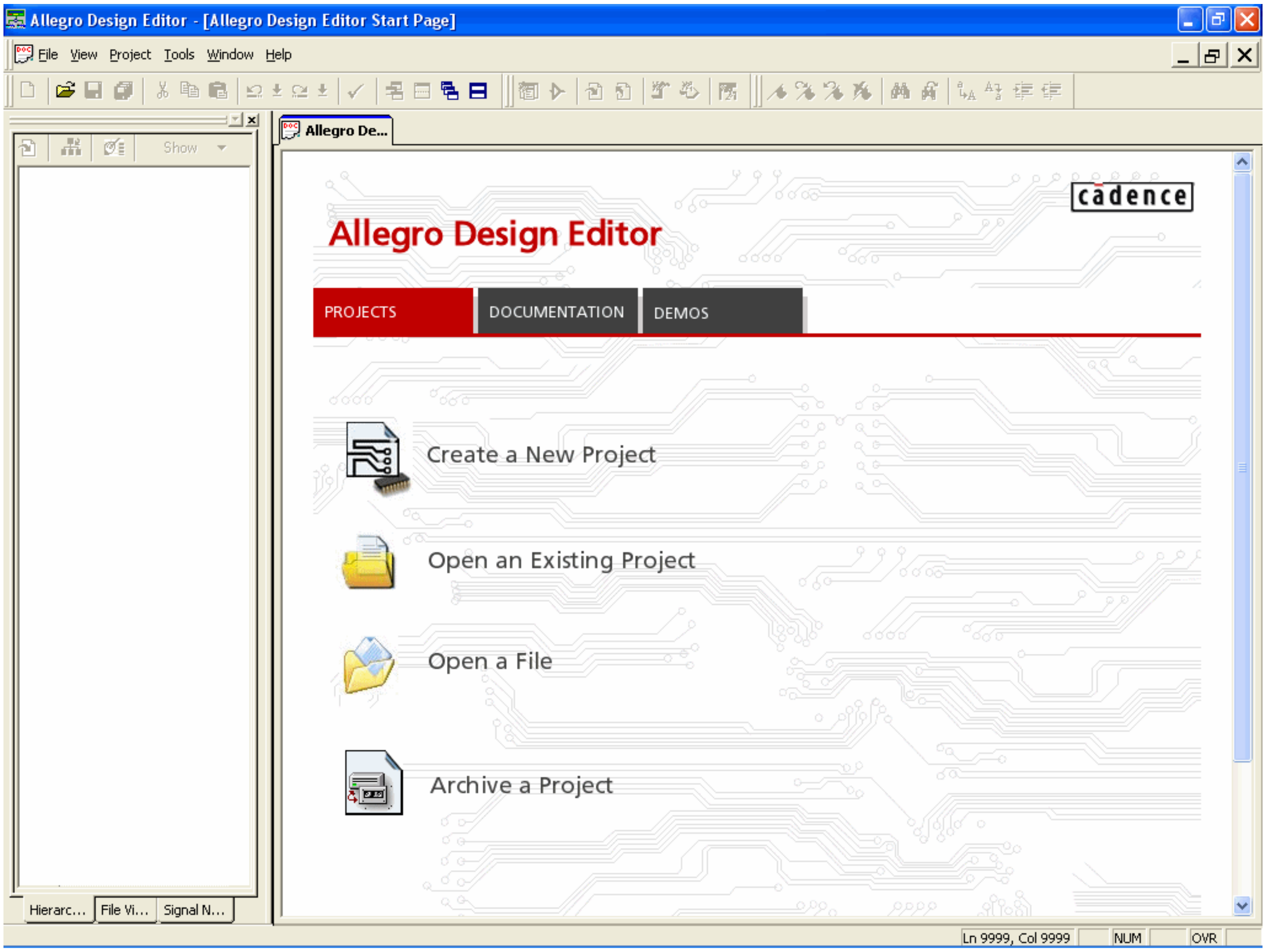
- From the Windows *Start* menu, choose *Programs – Allegro SPB – Design Editor*.
- From the Windows Start menu, choose *Run* to open the Run dialog box. Type `designstudio` and press *Enter*.
- Type the following command in the Windows command prompt:

  ```
  designstudio
  ```

To start Design Editor on Unix or Linux, open a terminal window and type the following command:

```
designstudio
```

The Design Editor Start Page appears.



Lesson 1-2: Creating a Project

Overview

In this lesson, you will learn to create a project in Design Editor.

Procedure

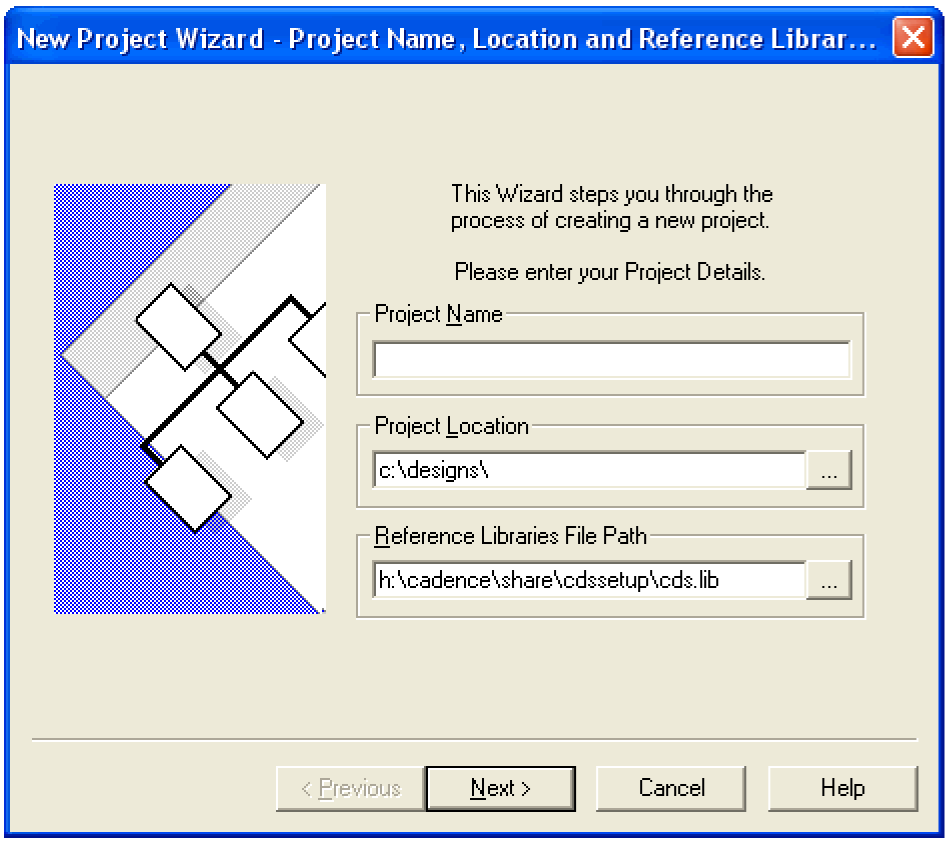
The New Project Wizard guides you through creating your project in Design Editor.

1. To access the New project wizard, do one of the following:

   ❑ Click the *Create a New Project* icon in the Design Editor Start page.

   ❑ Choose *File – New – Project*.

   The Project Name, Location and Reference Library Location page appears.



   The default project location is displayed as `c:\designs\` because the `TDD_START_PROJ_LOCATION` environment variable is set to `c:\designs`. For more information on the `TDD_START_PROJ_LOCATION` environment variable, see `Understanding the Sample Design Files` on page 14.

2. In the *Project Name* field, enter the project name as:

   `tutorial`

   The wizard will create a project file called `tutorial.cpm`.

Note: Use only lowercase letters, numbers and the underscore (_) character in project names.

3. The project name is automatically added in the *Project Location* field. This means that the project will be created in the `c:\designs\tutorial` folder.

   You can also use the browse button to specify the project location.

   Note: The name and location of the top level directory are defined by the *Project Location* field (not the *Project Name* field). Cadence recommends that you specify a top level directory name (in the *Location* field) that matches the project name. This helps in easily identifying the location of projects.

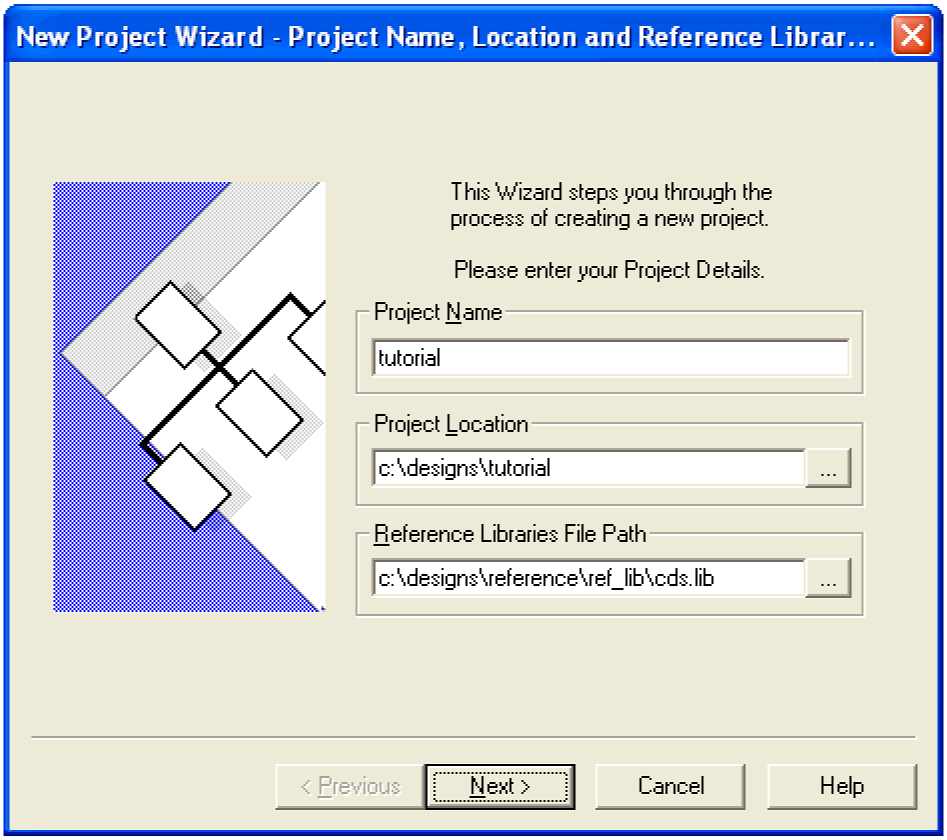
4. By default, the library definition file (`cds.lib`) in your Cadence installation directory is set as the reference library for the project. For this tutorial, we will use another reference library.

   In the *Reference Libraries File Path* field, enter the reference library file path as:

   `<your_work_area>\ref_lib\cds.lib`

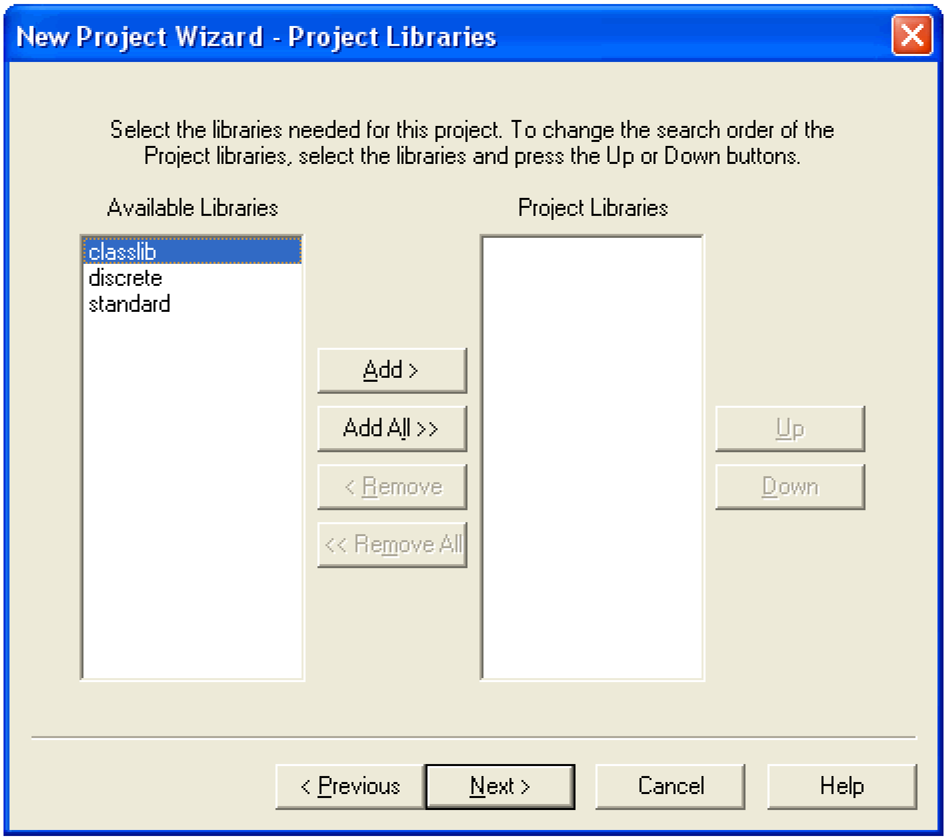
   Where <*your_work_area*> is the location where you have installed the sample design files for this tutorial.

This `cds.lib` file defines the path to the part libraries you will use in this tutorial.



5. Click *Next*.

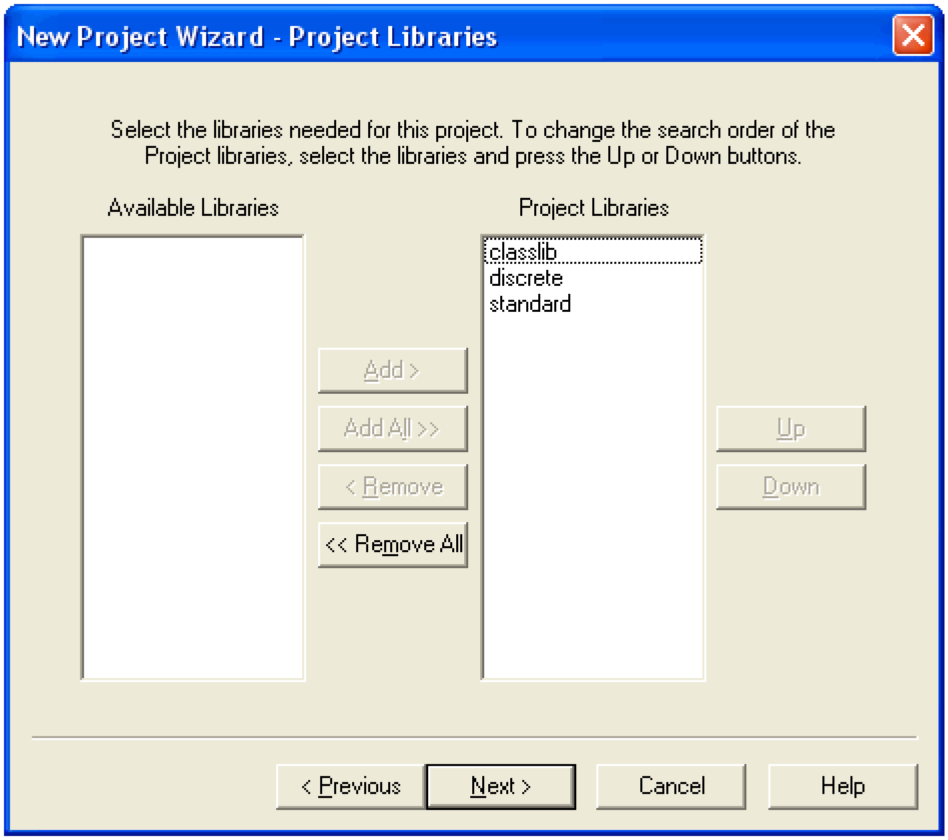
The Project Libraries page appears.



The reference libraries are displayed in the *Available Libraries* list. You can select the libraries you want to use in your project by adding them to the *Project Libraries* list. For this project you will add all the libraries to the Project Libraries list.

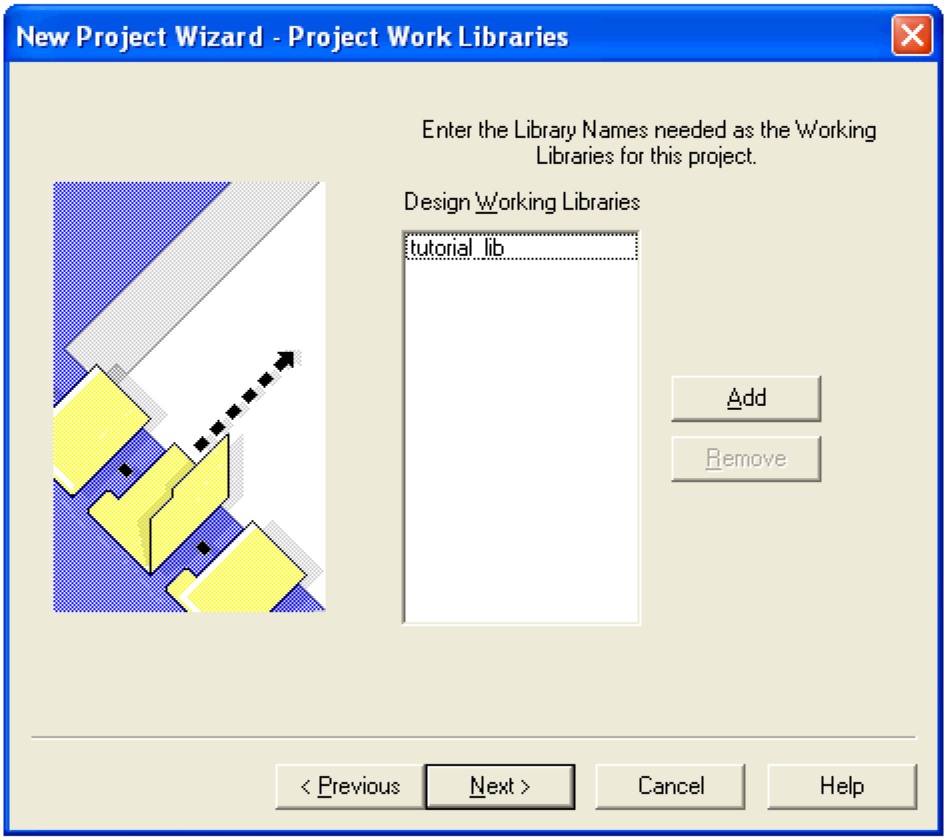
Note: If you do not add a library now, you can add it later using the *Library* tab in the Design Editor project setup. For more information, see Lesson 1-4: Setting Up Libraries for the Project on page 47.

6. Click the *Add All* button to add the reference libraries in the *Project Libraries* list.



7. Click *Next*.

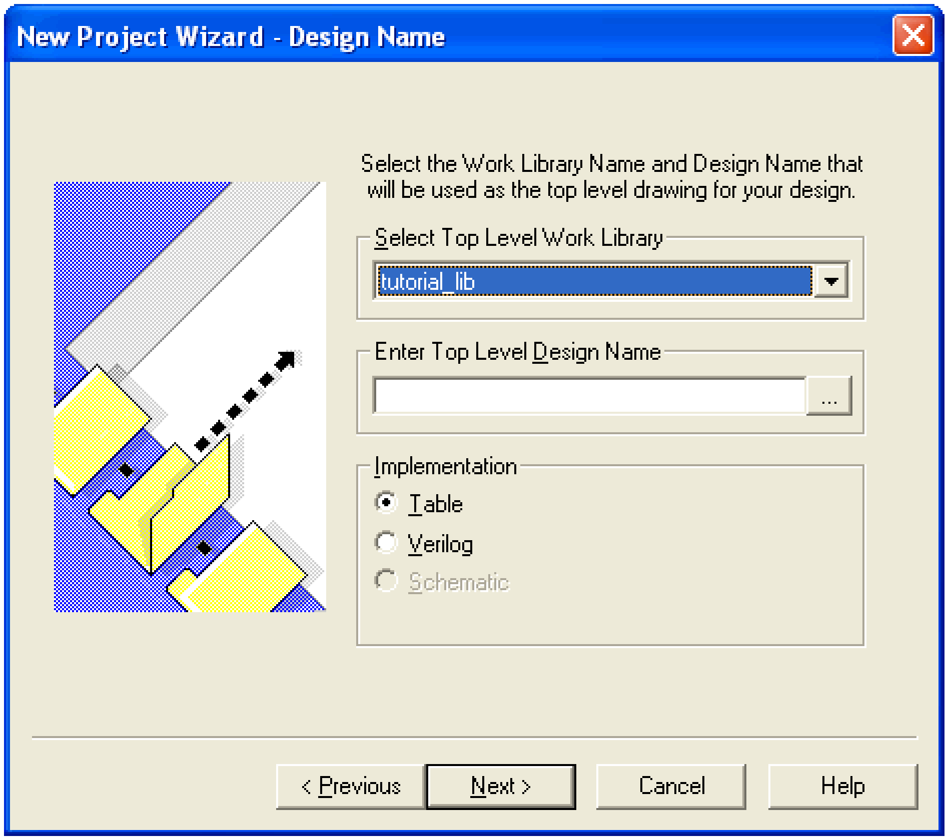
The Project Work Libraries page appears.



Here you can specify the working libraries in which the designs created by you will be stored. By default, the New Project Wizard creates a working library named `tutorial_lib` (`<`*`projectname`*`>_lib`). You can click the *Add* button to add more working libraries.

8. Click *Next*.

The Design Name page appears.



Here you can specify the name of the top-level or root design for the project and select the working library in which you want to create the top-level design.

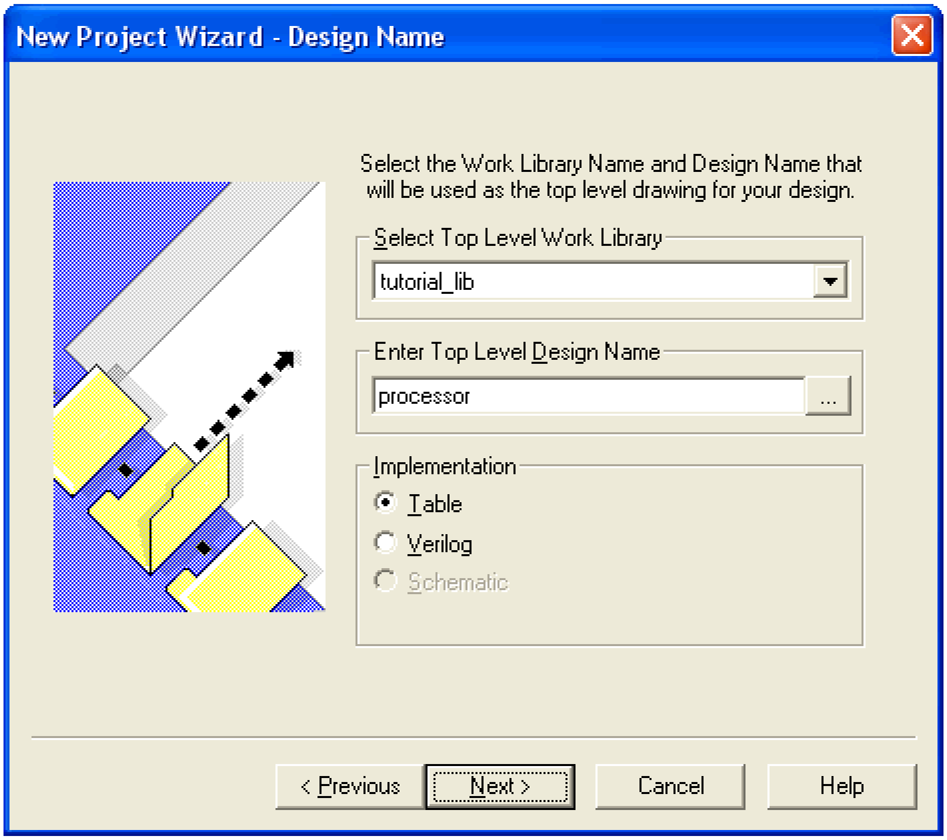
9. In the *Enter Top Level Design Name* field, enter the top-level design name for the project as:

`processor`

Note: Use only lowercase letters, numbers and the underscore (_) character in design names.

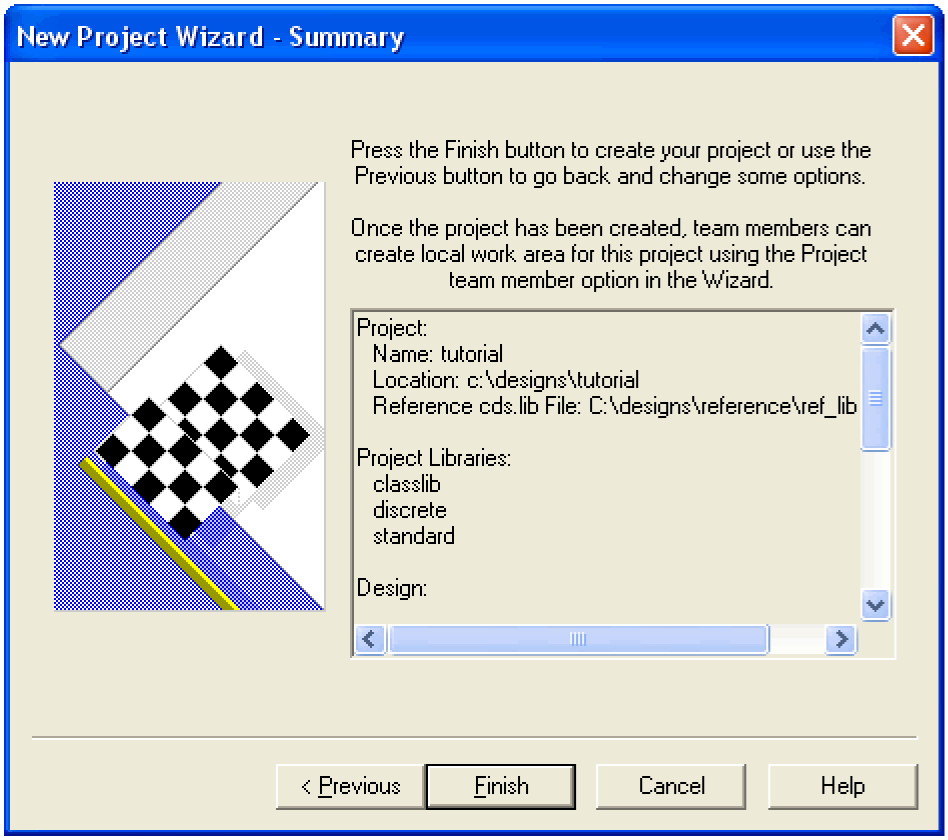
10. Design Editor lets you capture the top-level design using spreadsheets or using Verilog. For this project, you will capture the top-level design using spreadsheets.

Select the *Table* option.



11. Click *Next*.

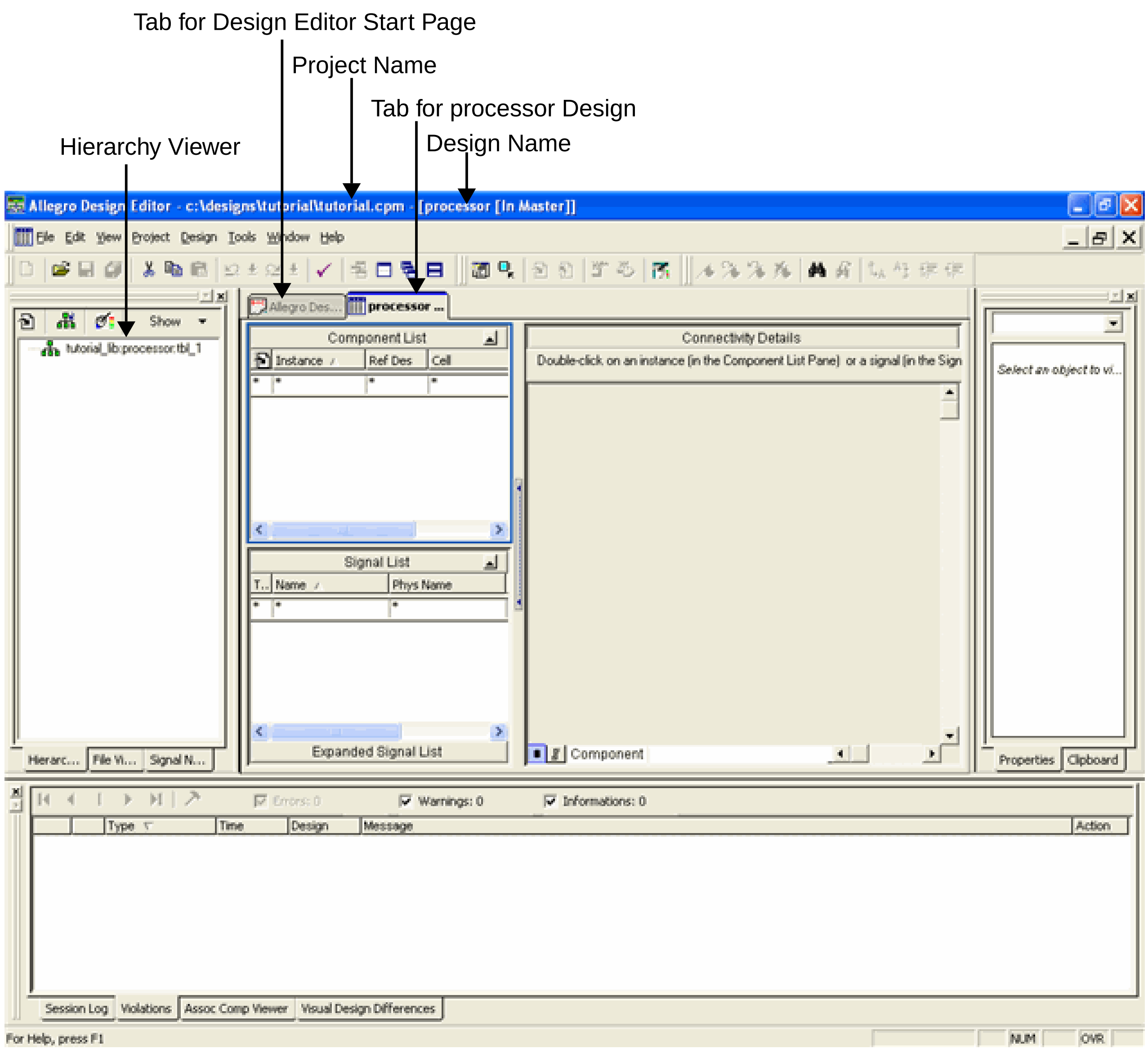
The Summary page appears.



Here you can review the settings for the project. To modify the settings, you can click *Previous* to go back to a previous page.

12. Click *Finish* to create the project.

After the project is created, it is automatically opened in Design Editor.



The project name and design name are displayed in the Design Editor title bar.

The Hierarchy Viewer displays the name of the top-level or root design in the `library:cell:view` format as:

Where `tutorial_lib` is the working library in which the `processor` design is saved and the `tbl_1` view indicates that the `processor` design is a spreadsheet based design.

Note that Design Editor displays the `processor` design in a new tab. You can view the Design Editor Start page by clicking the tab for the Design Editor Start page.

13. Use the Windows Explorer or UNIX or Linux terminal window to view the contents of the `tutorial` project directory.

Summary

You now know how to create a project in Design Editor. You also learned the naming conventions for project names and design names.

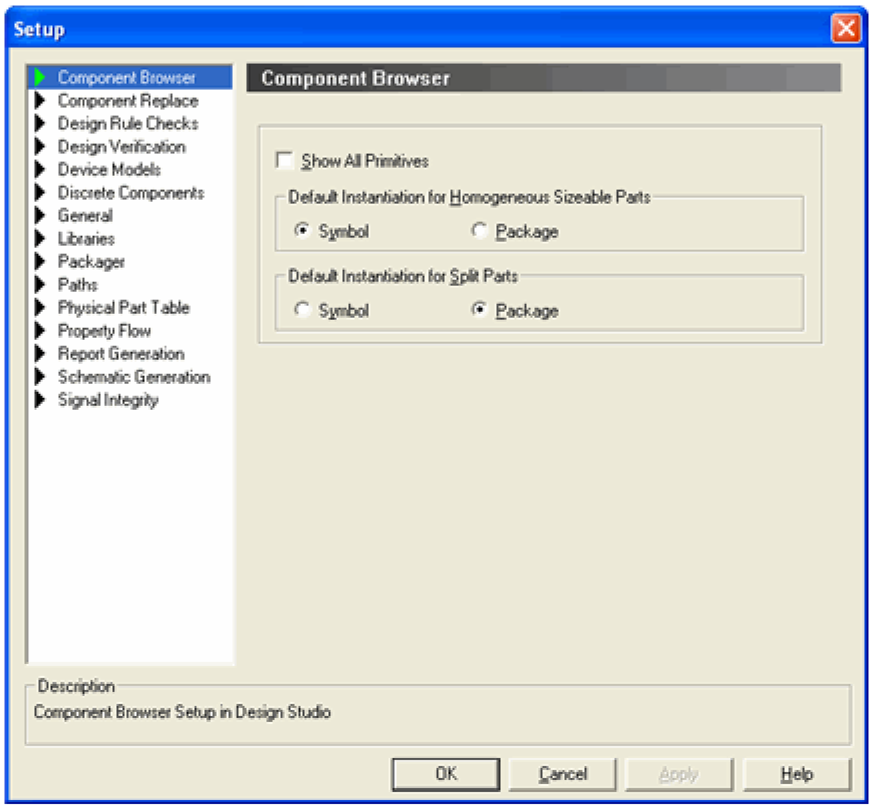
For More Information

See Working with Projects chapter of *Allegro Design Editor User Guide*.

Lesson 1-3: Setting Up the Project

Overview

After creating your project, you might want to change the default settings for your project. You can use the Setup dialog box to specify the settings for the project.



In this lesson, you will learn to include physical part table (`.ptf`) files for the project.

The Physical Part Table (.ptf) file stores the packaging properties for a part in the library. This file contains information about parts such as package types, manufacturers, part numbers and any custom properties. Each physical part must have an entry in the `.ptf` file in order to package properly. Part table files can be located in the `part_table` view of cells in a library or outside the library structure. For more information on the structure of part libraries, see Figure 1-1 on page 24.

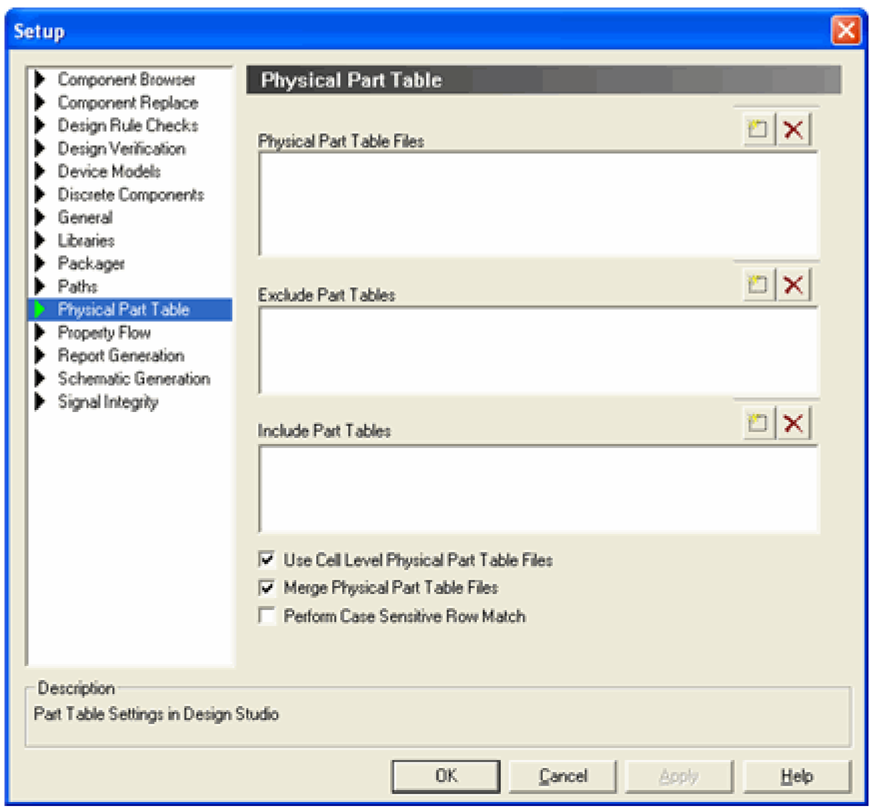
To access the information contained in part table files, you must include them in your project. By default, the part table files located in the `part_table` view of cells in a library are included in your project. If a part table file is located outside the `part_table` view of cells in a library, you must include the file in your project.

Procedure

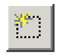
1. Choose *Project – Setings*.

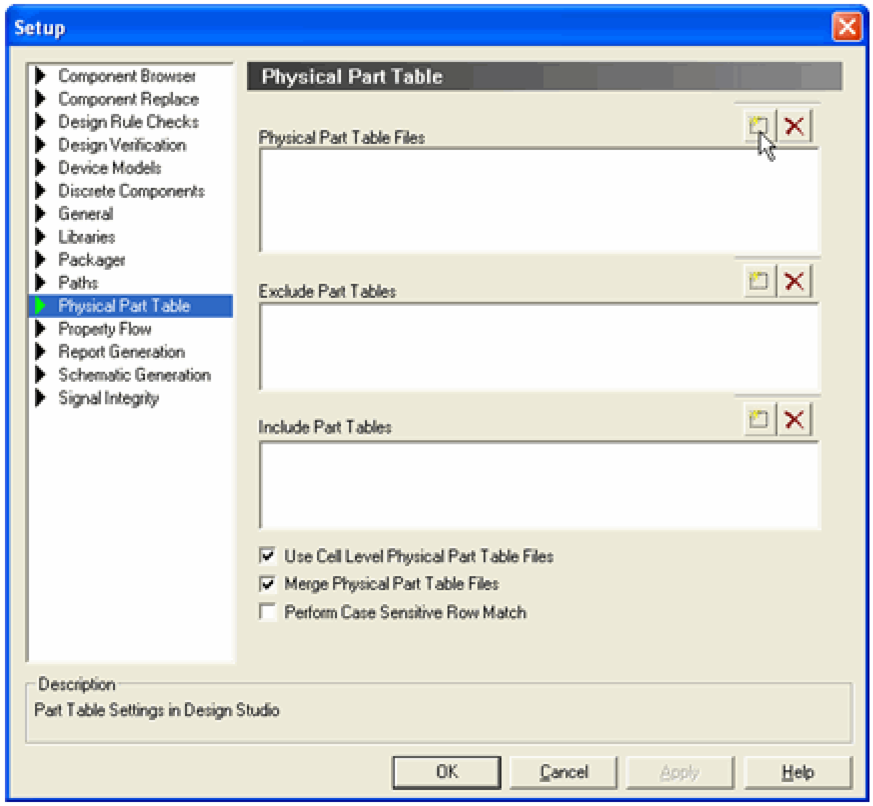
   The Setup dialog box appears.

2. Click the *Physical Part Table* tab.

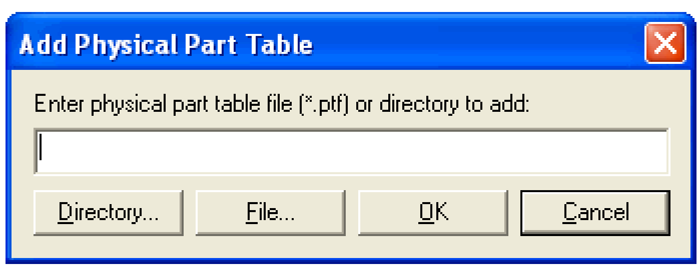


By default, the *Use Cell Level Physical Part Table Files* check box is selected. This indicates that the part table files located in the `part_table` view of cells in the libraries added for your project are automatically included in the project.

3. Click the  button above the *Physical Part Table Files* list.



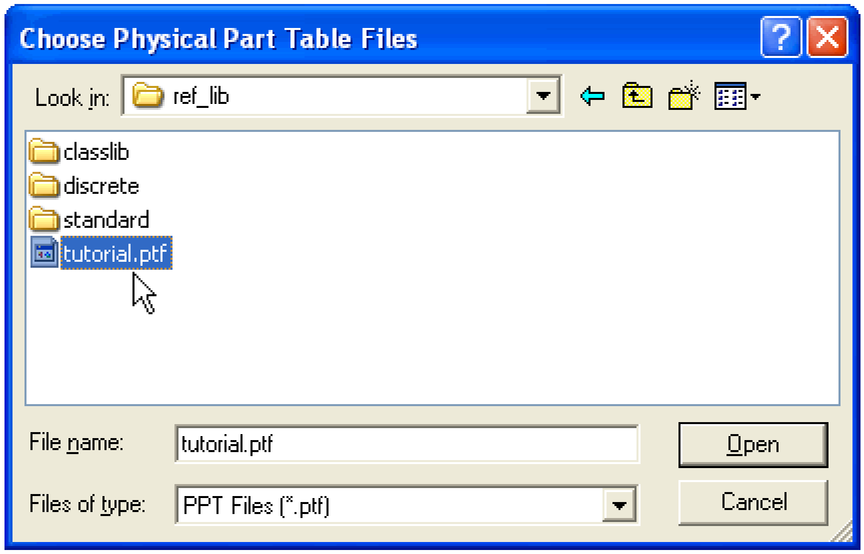
The Add Physical Part Table dialog box appears.



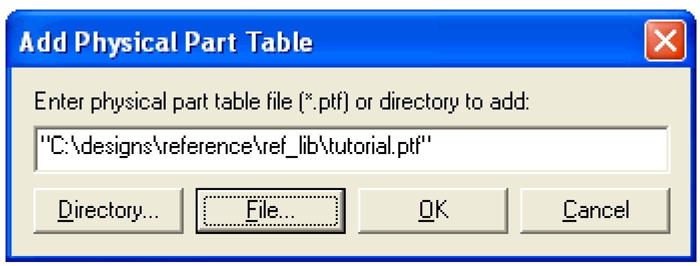
4. Click *File*.

The Choose Physical Part Table Files dialog box appears.

5. Select the `tutorial.ptf` file located at <*your_work_area*>`\reference\ref_lib` and click *Open*.

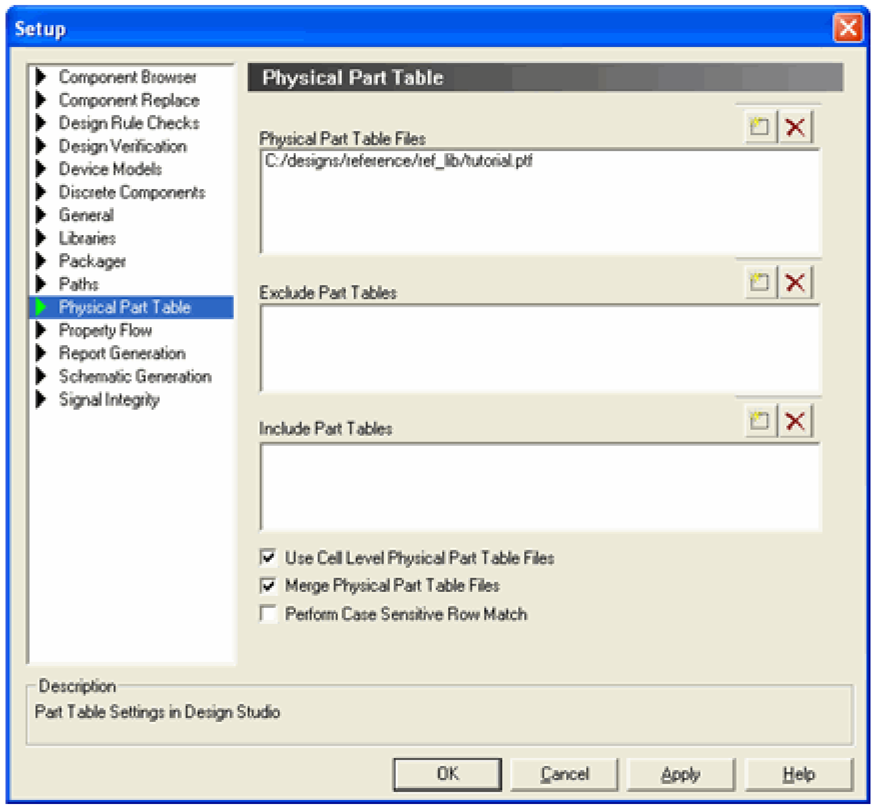


6. The Add Physical Part Table dialog box displays the path to the `tutorial.ptf` file.



7. Click *OK*.

The *Physical Part Table Files* list in the *Physical Part Table* tab displays the path to the `tutorial.ptf` file.



8. Repeat steps 3 to 7 to add the `mech.ptf` file located at:

   `<`*`your_work_area`*`>\reference\cdssetup\`

   in the *Physical Part Table Files* list.

   The *Include Part Tables* list allows you to specify the part table files listed in the *Physical Part Table Files* list that should be used during packaging.

9. Click the  button above the *Include Part Tables* list.

   The Include Physical Part Table dialog box appears.

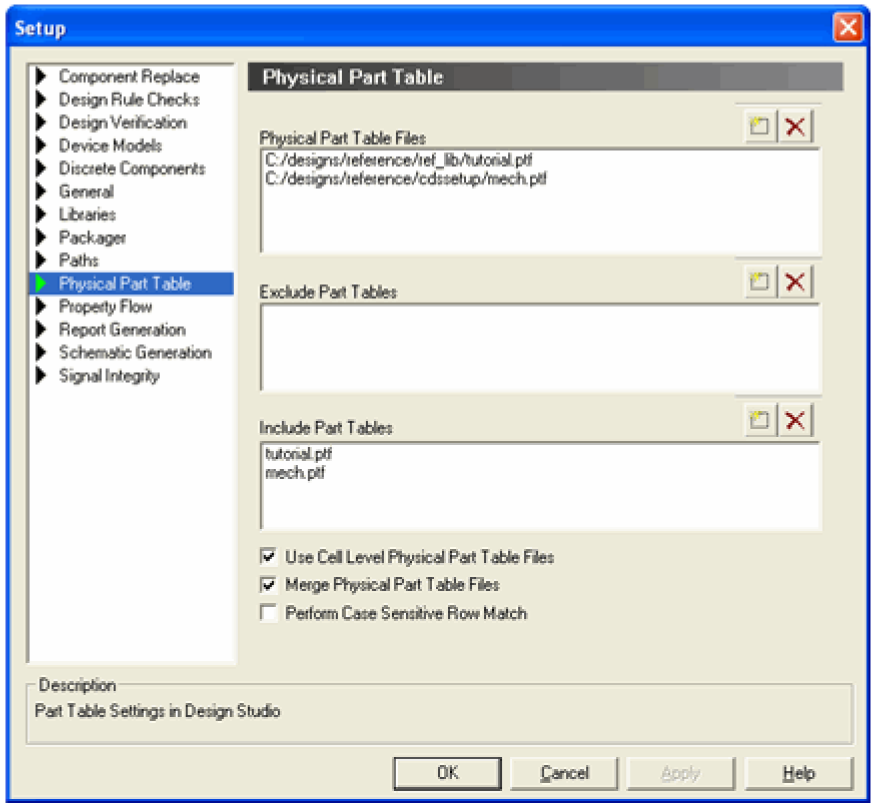
10. In the *Enter name of part table to include* field, enter:

    `tutorial`

    Note: Enter only the name of the part table file, without the `.ptf` extension.

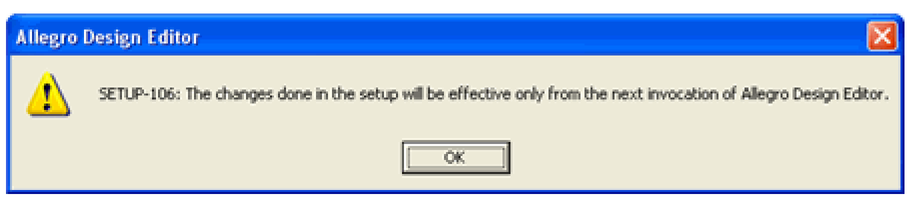
11. Click *OK*.

12. Repeat steps 9 to 11 to add `mech` in the *Include Part Tables* list.



13. Click *OK* to save the changes and close the Setup dialog box.

The following message box appears, indicating that the change will be effective only if you reopen the project in Design Editor.



14. Click *OK*.

Summary

You now know how to open the Setup dialog box to specify the options for setting up your project. You also learned about physical part table files and how to include them in your project.

For More Information

For more information on physical part table (`.ptf`) files, see the *Allegro Design Entry HDL Libraries Reference*.

Lesson 1-4: Setting Up Libraries for the Project

Overview

You can add only the parts existing in the libraries setup for your project. In this lesson, you will learn to setup libraries for the project.

Before you setup a library for your project, you must define the library in the `cds.lib` file for your project. In this lesson, you will include the library named `analog` in the `cds.lib` file for your project and then setup the library for your project.

Procedure

1. Choose *File – Exit* to close Design Editor.
2. Open the `cds.lib` file located at <*your_work_area*>`\tutorial` in a text editor.

   The `cds.lib` file has the following entries:

   ```
   DEFINE tutorial_lib ./tutorial_lib
   INCLUDE <your_work_area>/reference/ref_lib/cds.lib
   ```

   The first line defines the working library named `tutorial_lib` and the second line specifies the path to the reference `cds.lib` file located at:

   ```
   <your_work_area>/reference/ref_lib
   ```

   Use the `DEFINE` command to define a library. The library name is specified first, followed by the path to the library directory. The path can be absolute or relative to the location of the `cds.lib` file. Except for the path names, the `cds.lib` file is not case sensitive.

Use the `INCLUDE` command to load another library definition (`cds.lib`) file. For example, you can include a `cds.lib` file that defines a list of company-generated libraries, or you can include the `cds.lib` file being used in another project.

3. Add the following line in the `cds.lib` file:

```
DEFINE analog <your_work_area>/reference/ref_lib/analog
```

The above syntax defines the `analog` library located at `<your_work_area>/reference/ref_lib/` in the `cds.lib` file.

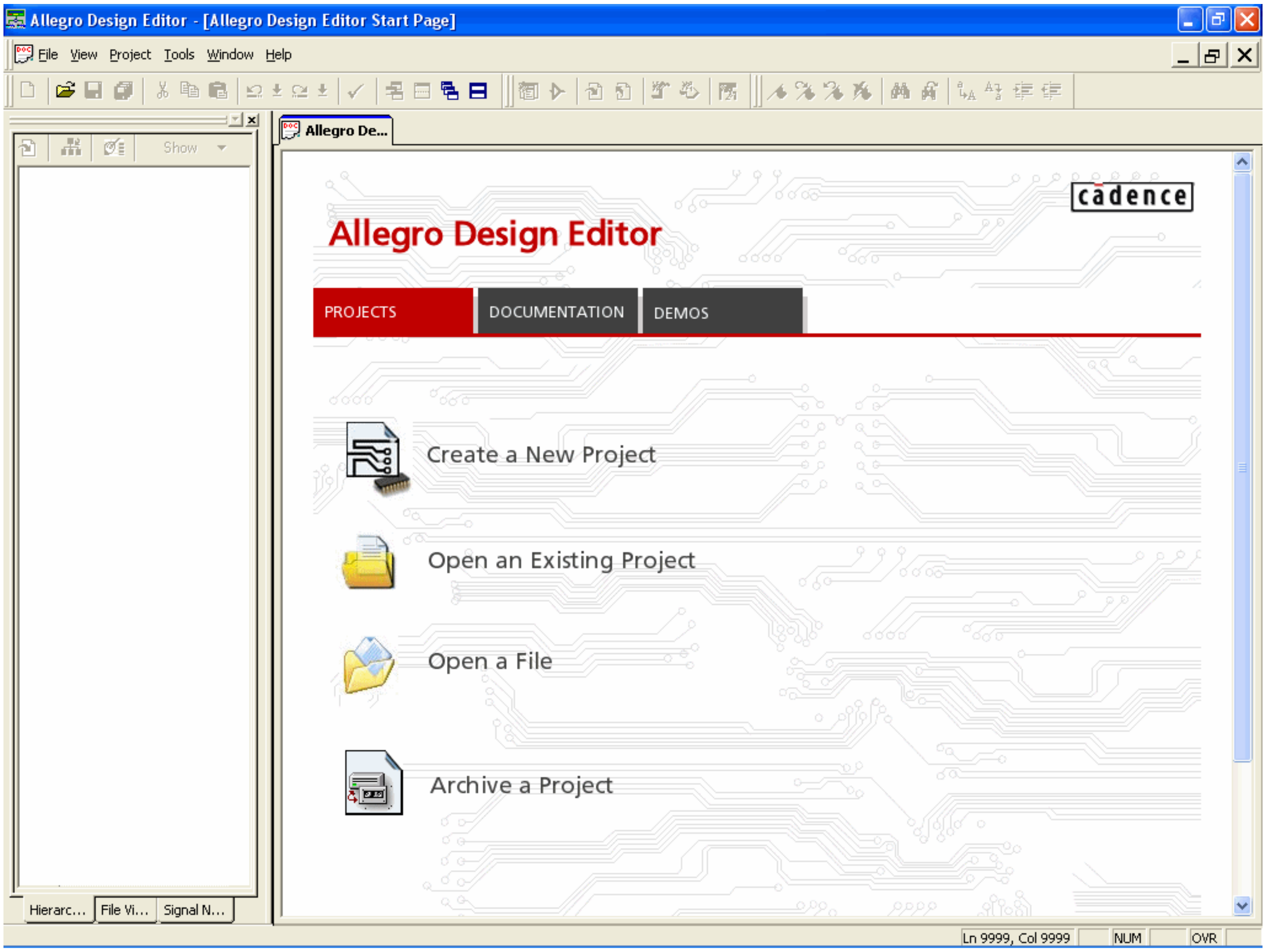
Note the following:

- ❑ Use lowercase library names only in the `cds.lib` file. Do not use mixed or uppercase names, or special characters except the underscore character.
- ❑ Use forward slashes in paths as they will work on Windows, UNIX and Linux platforms. Backward slashes and absolute paths will need to be modified if the project is transferred between Windows, UNIX and Linux platforms.

4. Start Design Editor.

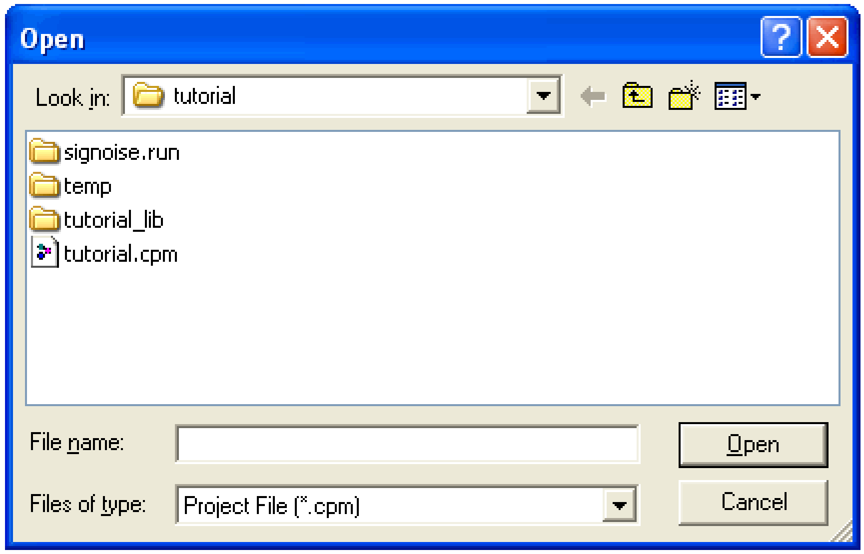
For more information on starting Design Editor, see Lesson 1-1: Starting Design Editor on page 28.

The Design Editor Start page appears.



5. Click the *Open an Existing Project* icon in the Start page.

The Open dialog box appears.

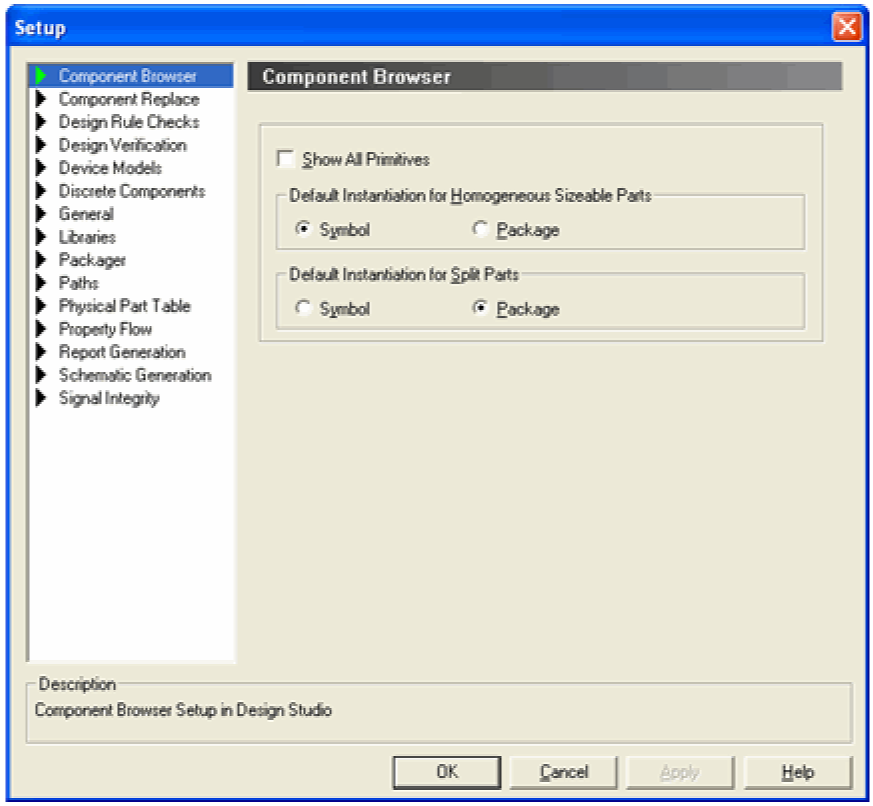


6. Select the `tutorial.cpm` file located at <*your_work_area*>\tutorial\ and click *Open*.

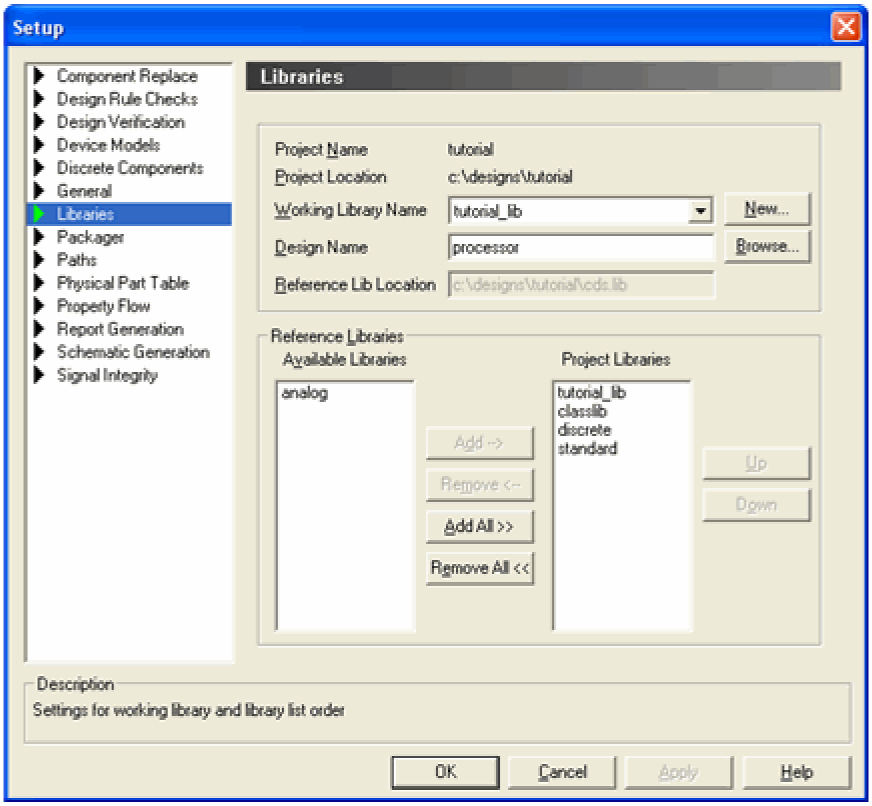
The project is opened in Design Editor.

7. Choose *Project – Setings*.

The Setup dialog box appears.

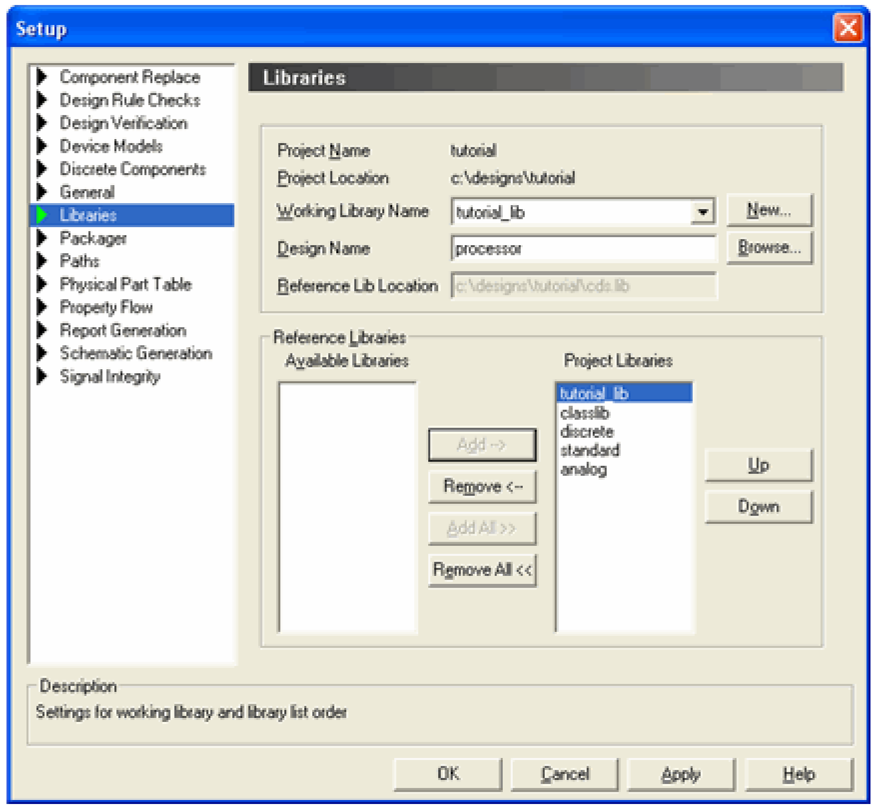


8. Click the *Libraries* tab.



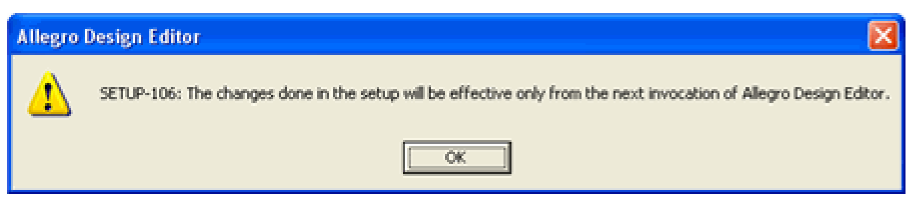
Note that the `analog` library is displayed in the *Available Libraries* list.

9. Select the `analog` library and click the *Add* button to add the `analog` library to the *Project Libraries* list.



10. Click *OK*.

    The following message box appears:



11. Choose *File – Exit* to close Design Editor.
12. Start Design Editor and open the project.

    The `analog` library is now setup for the project.

Summary

You now know how to setup libraries for the project. You also learned the following:

- Syntax used in the `cds.lib` file.
- How to exit Design Editor.
- How to open a project in Design Editor.

For More Information

For more information on the `cds.lib` file, see the Allegro Design Entry HDL Libraries Reference.

Module 2: Working with Components and Connectivity

Prerequisite

If you have not completed all the lessons in Module 1: Working with Projects, you must open the `tutorial.cpm` project located at `<your_work_area>\modules\connectivity\tutorial` in Design Editor and perform the steps described in this module.

For more information, see Understanding the Sample Design Files on page 14.

Lessons

This module consists of the following lessons:

Multimedia Demonstration

Click the link below to view a Flash-based multimedia demonstration of this module.

Working with Components and Connectivity

Completion Time

3 hours for written lessons

30 minutes for multimedia demonstrations

Overview

Once you have created your project and specified the settings for your project, you can proceed with capturing the logic for your design.

One of the primary tasks in design capture involves adding components in the design and capturing the connectivity information for the design by connecting component pins to signals.

The following lessons tell you how to add components in the design and capture connectivity information.

Lesson 2-1: Adding Components

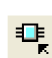
Overview

In this lesson, you will learn to add components in the design in Design Editor.

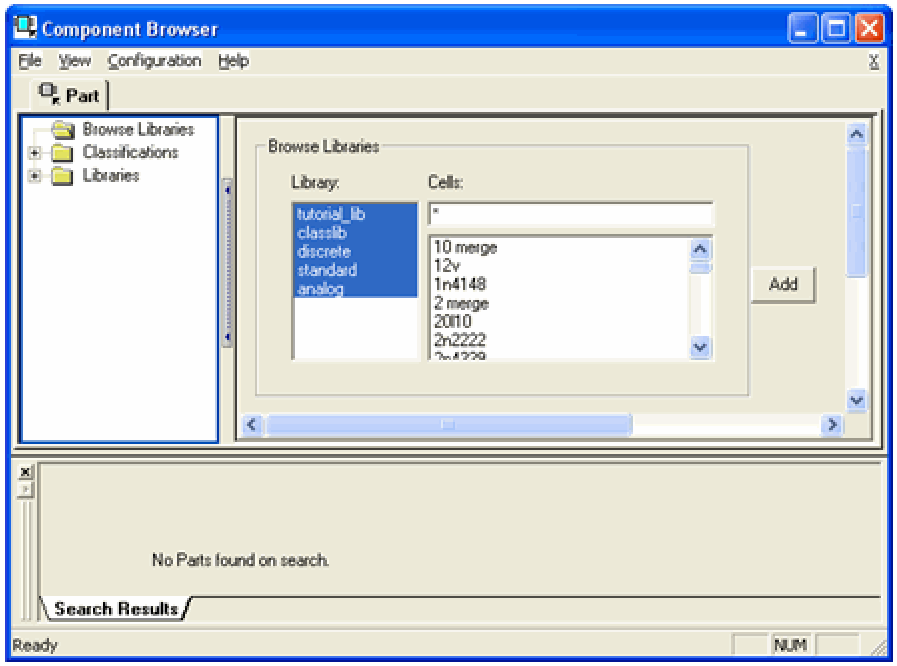
You use the Component Browser to choose components from project libraries and add them in the design. The components added in the design are displayed in the Component List in Design Editor.

For more information on setting up libraries for your project, see Lesson 1-4: Setting Up Libraries for the Project on page 47.

Procedure

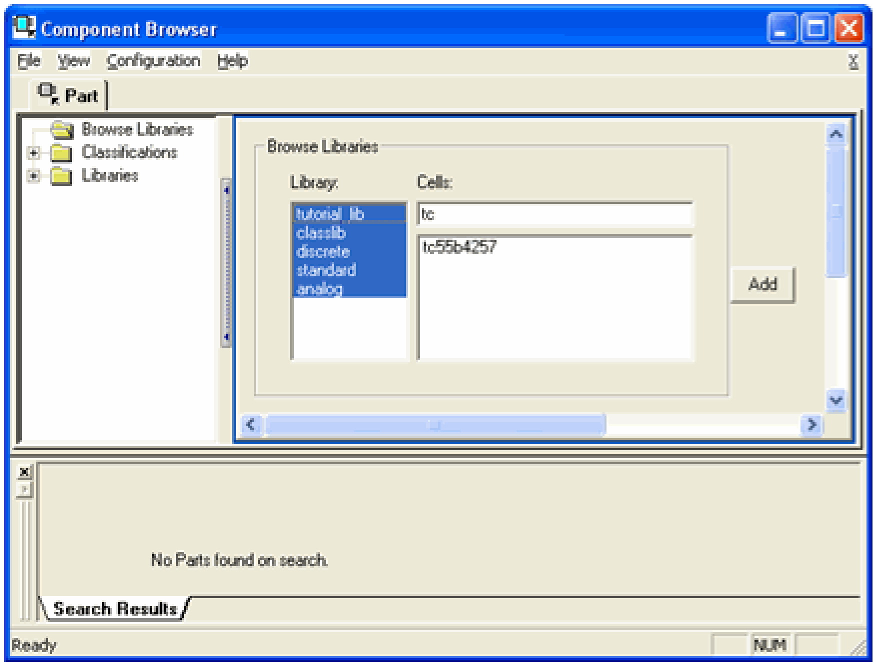
1. To open the Component Browser, do one of the following:
   - ❑ Choose *Design – Add Component*.
   - ❑ Click the  toolbar button.

The Component Browser appears.



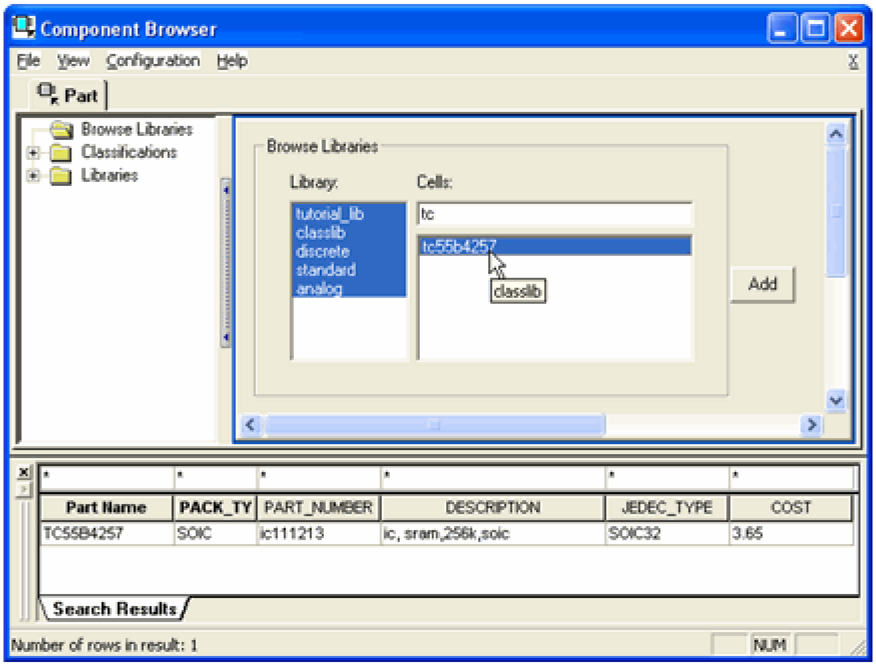
By default all the libraries are selected. All the cells in the selected libraries are displayed in the *Cells* list. If you know the name of the component you want to add, type it partially in the text box or use wildcards. This will display all matching components in the *Cells* list.

2. Type `tc` in the text box.



The memory component `tc55b4257` is displayed in the *Cells* list.

3. Select the `tc55b4257` component in the *Cells* list.

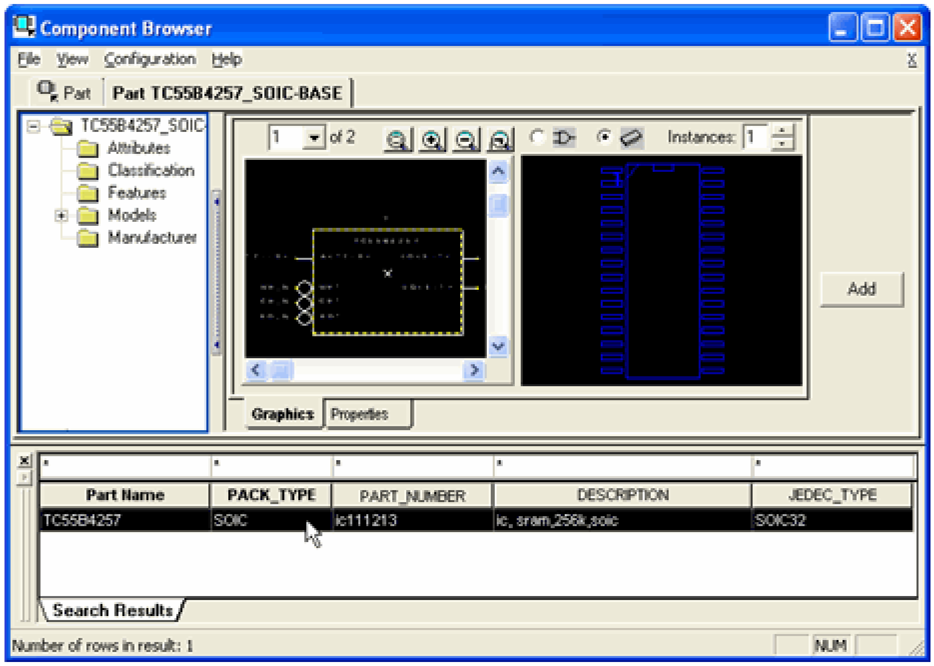


The part table rows available for the component are displayed in the *Search Results* pane.

Note: If you place the mouse pointer on a component in the *Cells* list, the library in which the component exists is displayed as a tooltip. In the above figure, the tooltip indicates that the `tc55b4257` component exists in the `classlib` library.

4. Select the physical part in the *Search Results* pane.

The component details are displayed in a new details tab named *Part TC55B4257_SOIC-BASE*.

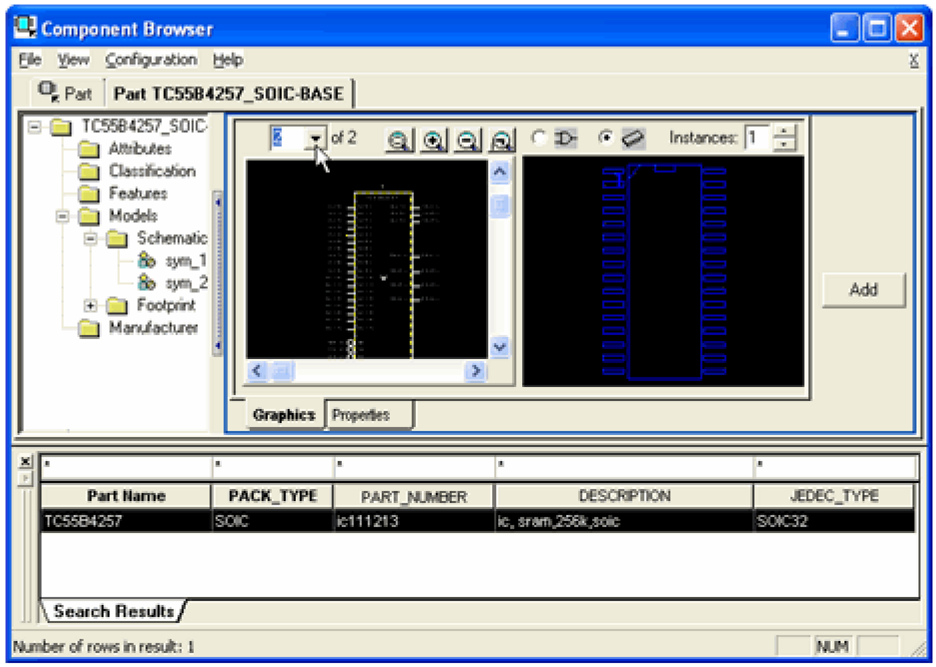


The default symbol version for the component is displayed in the symbol viewer. The footprint for the component is displayed in the footprint viewer.

Note: If the footprint for the component is not displayed in the footprint viewer, check whether the `PADPATH` and `PSMPATH` Allegro environment variables are set. For more information, see Setting Environment Variables on page 15.

To view a different symbol version select the version number in the version drop-down list.

5. Select 2 in the version drop-down list to view the second symbol version for the `tc55b4257` component.



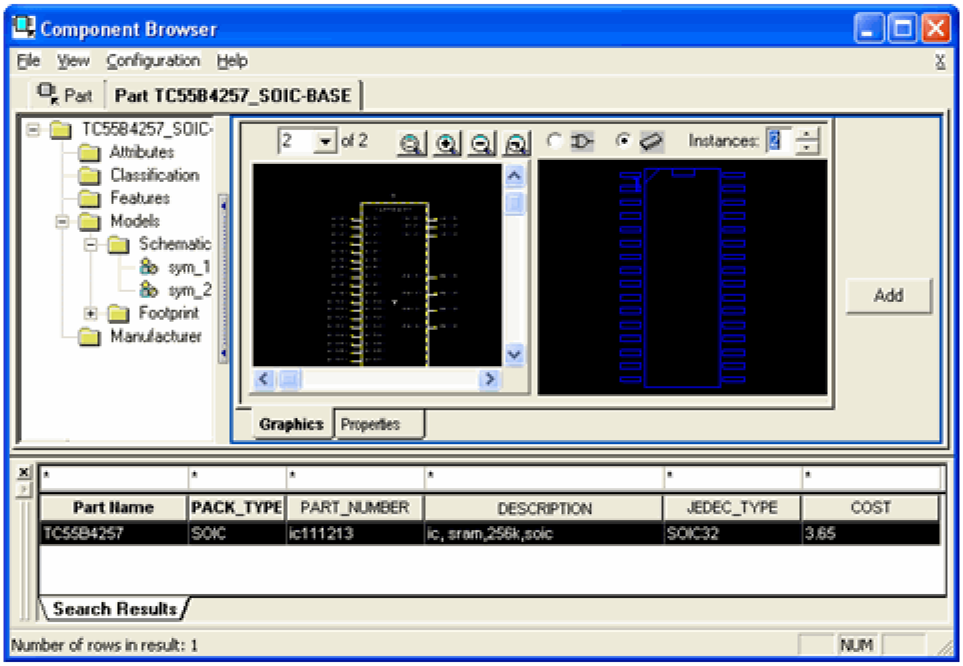
You can do one of the following:

- ❑ Select the option to instantiate the `tc55b4257` component in the design by adding every symbol version of it. The number of instances comprising the component is equal to the number of symbol versions for the component.
- ❑ Select the option to add the `tc55b4257` component as a package (one instance that represents the complete component).

You will now add the `tc55b4257` component as a package.

6. Select the option.

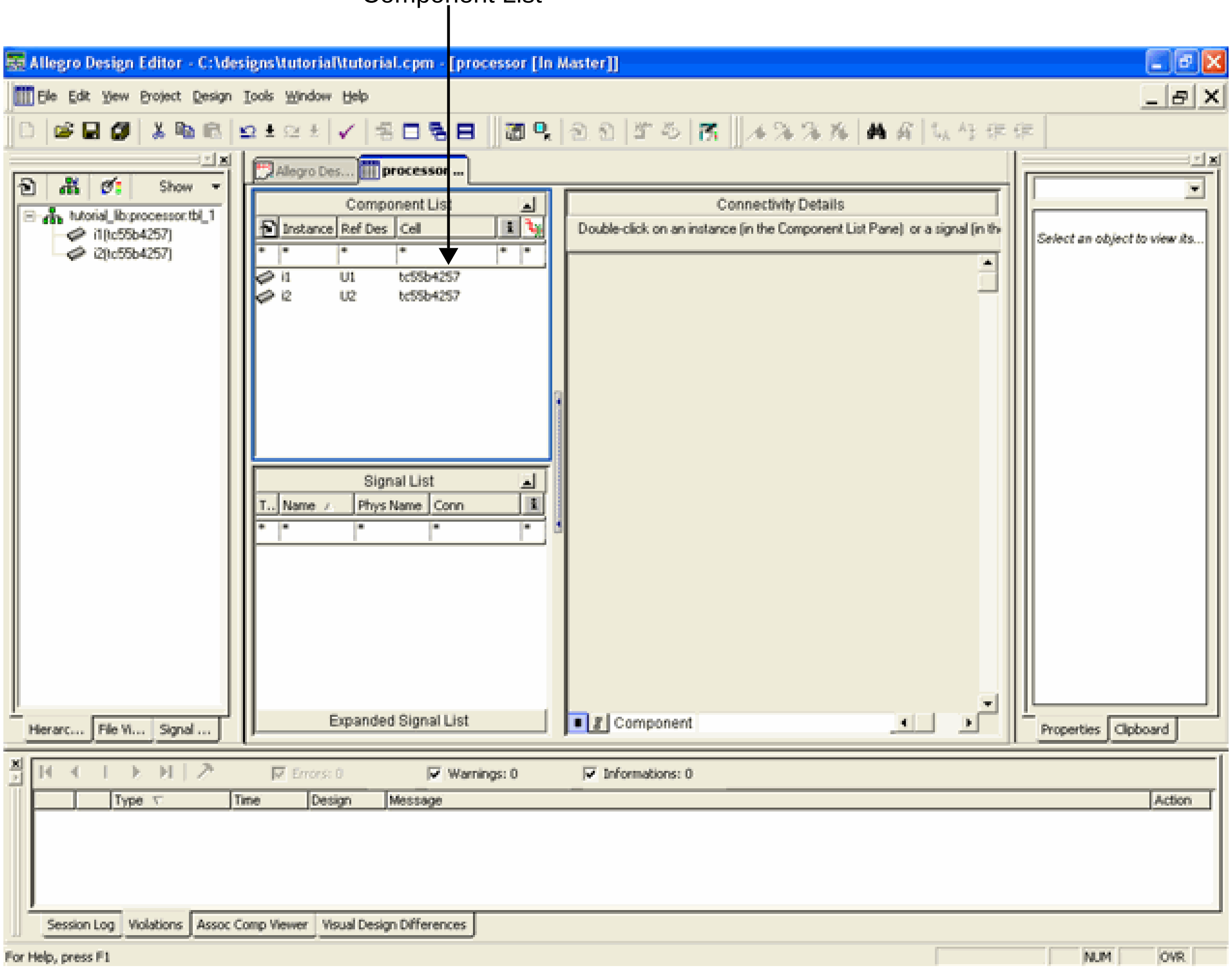
7. Type 2 in the *Instances* field.



8. Click the *Add* button.

Two instances of the `tc55b4257` component are added in the design and displayed in the Component List in Design Editor.

Component List

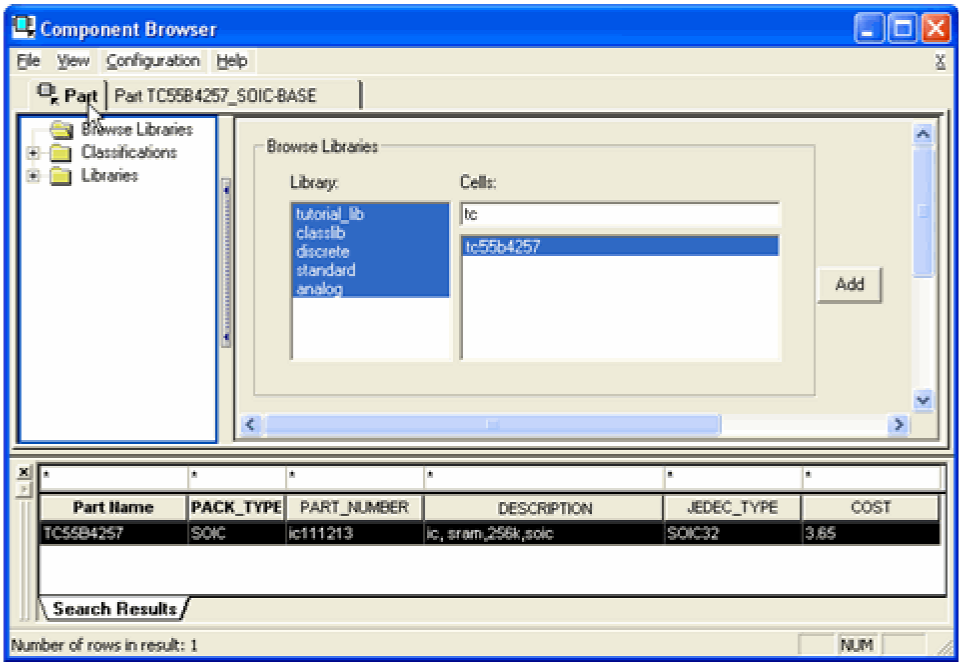


The Component List lets you work with the components in the design. Each row in the Component List corresponds to a component in the current design.

When you add a component in the design, Design Editor automatically packages the component and assigns a reference designator to the component. Design Editor also assigns an instance name to the component. In the above figure, note that the `tc55b4257` components displayed in the Component List have been assigned the reference designator `U1` and `U2` and the instance names `i1` and `i2`.

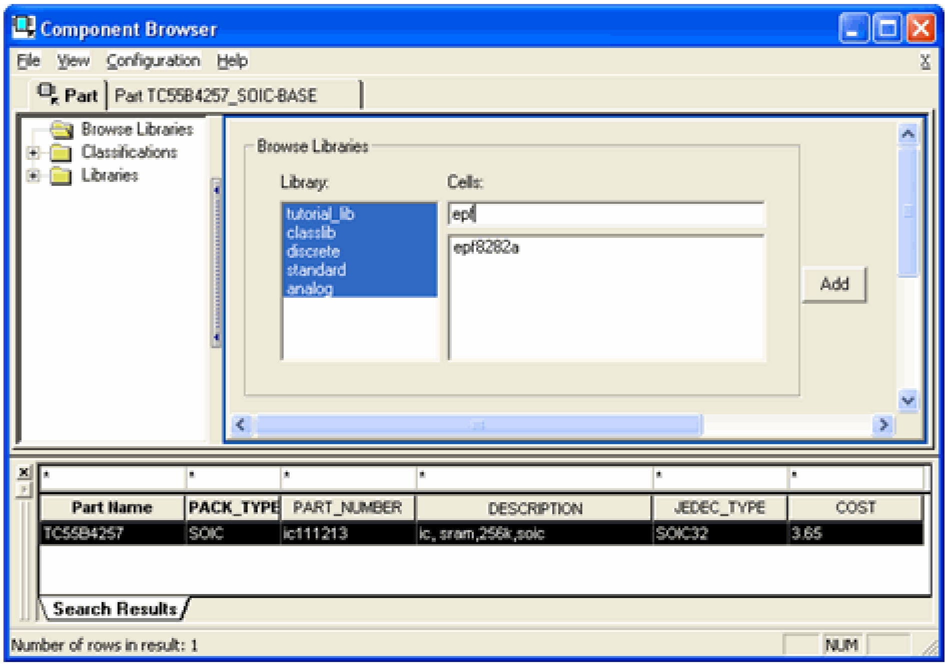
Note: You can modify the instance name and reference designator of the component.

9. Click the *Part* tab in the Component Browser.



10. Type `epf` in the text box.

The FPGA component `epf8282a` is displayed in the *Cells* list.

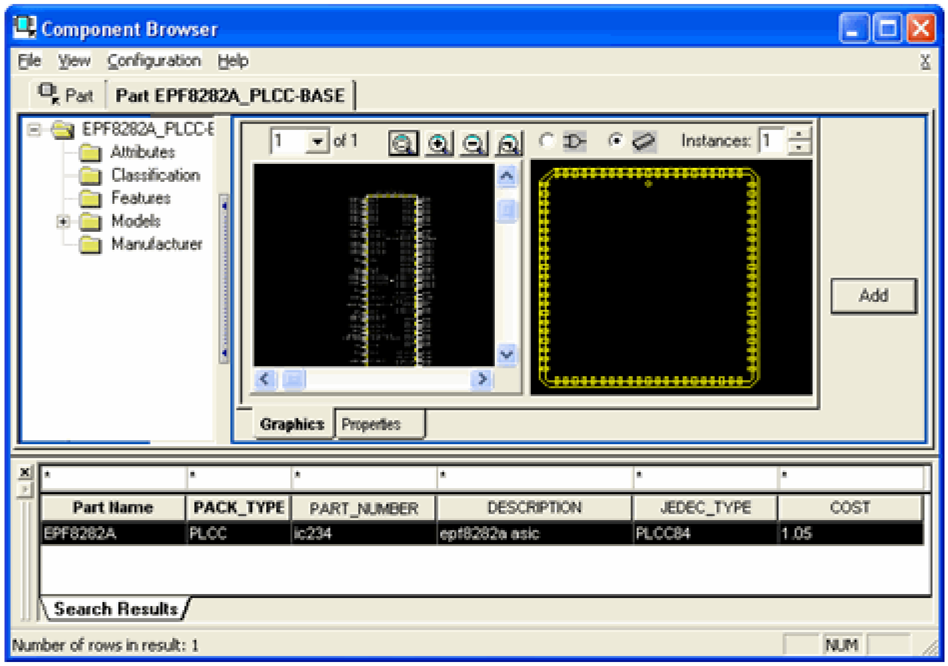


11. Select the `epf8282a` component in the *Cells* list.

12. Select the physical part in the *Search Results* pane.

The component details are displayed in a new details tab named *Part EPF8282A_PLCC-BASE*.

Enter `1` in the *Instances* field.



13. Click *Add*.

The `epf8282a` component is added in the design and displayed in the Component List.

14. Close the Component Browser.

15. Choose *File – Save* to save the design.

Summary

You now know how to use the Component Browser to add components in the design. You also learned the following:

- You can add a symbol version of the component or add the component as a package (one instance that represents the complete component).
- Design Editor automatically packages the components you add in your design and assigns a reference designator to the components.

For More Information

See:

Working with Components and Connectivity chapter of *Allegro Design Editor User Guide*.

Lesson 2-2: Adding Signals

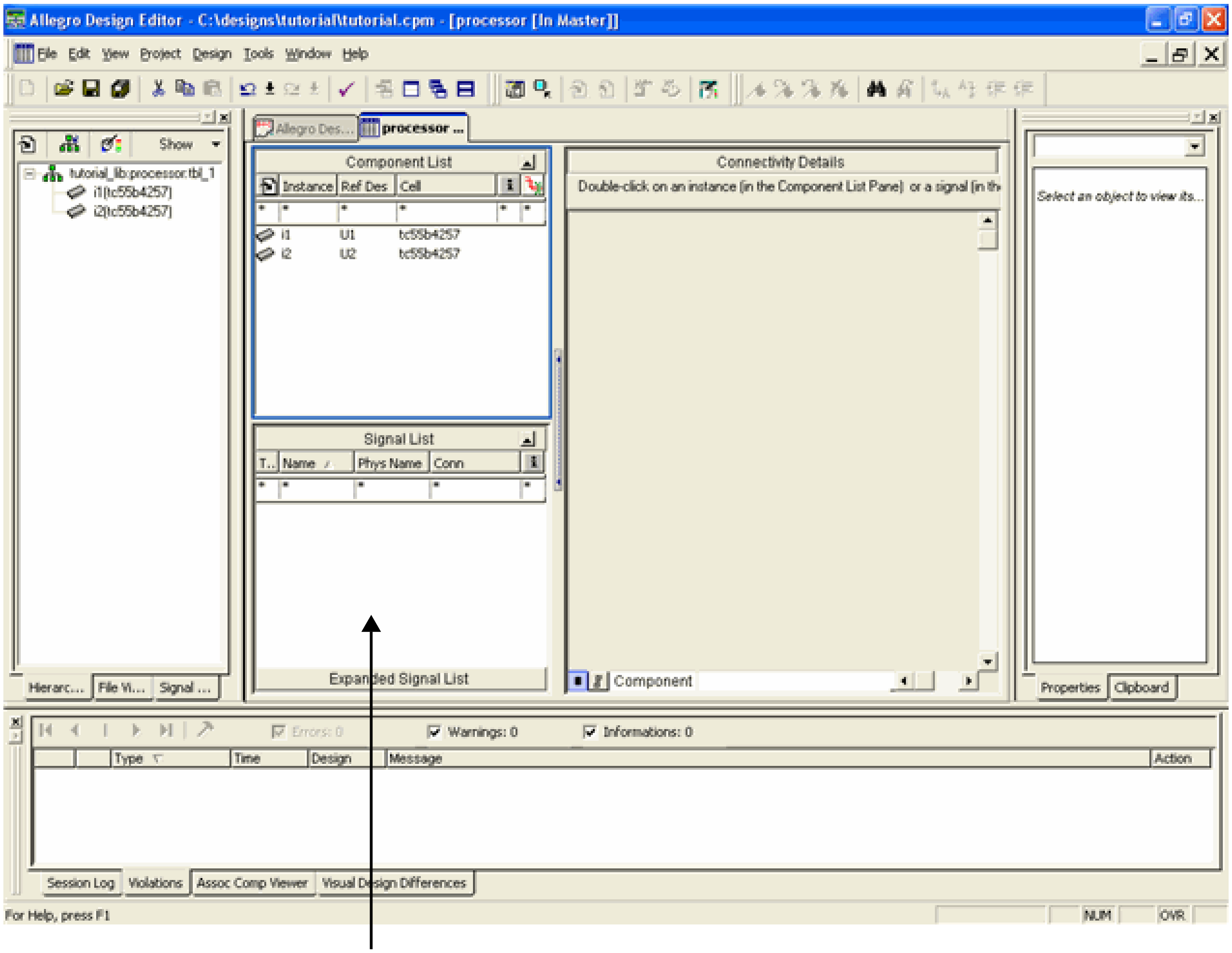
Overview

The Signal List in Design Editor lets you work with signals in the design.

In this lesson, you will learn to use the Signal List to add signals in the design.

Procedure

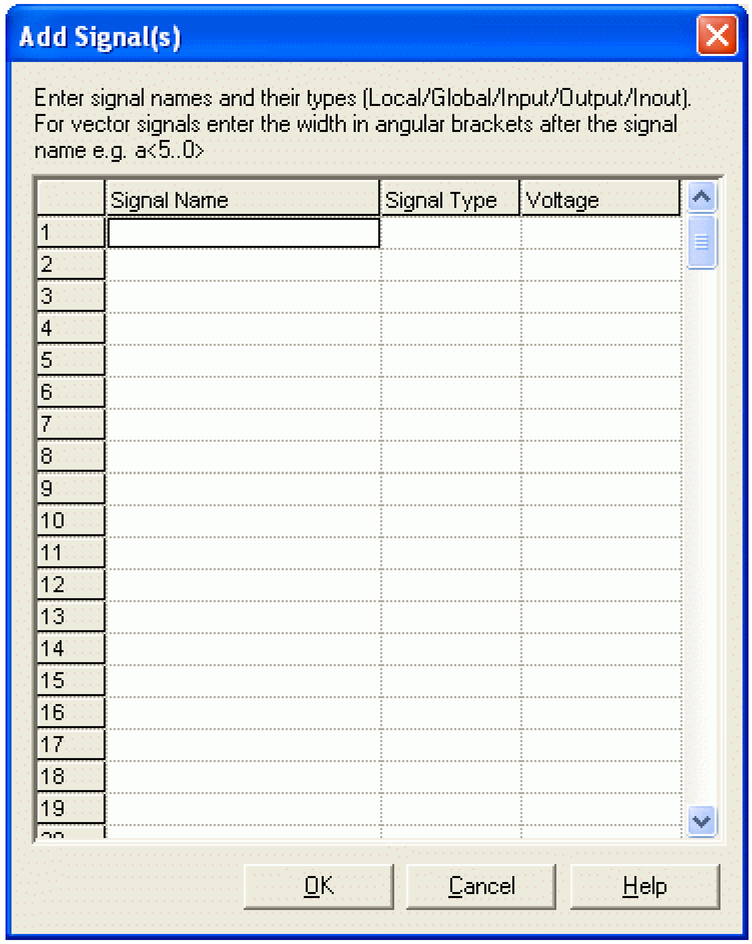
1. Click on the Signal List pane.



Signal List

2. Choose *Design – Add Signal*.

The Add Signal(s) dialog box appears.



Here you can specify the signal name, its scope and voltage. The scope of a signal can be `LOCAL`, `GLOBAL`, `IN`, `OUT` or `INOUT`. Voltage is required to identify DC nets in the design.

DC nets are required for connecting to associated components (terminations, bypass capacitors and pullup/pulldown) in the design. For more information on working with associated components, see Module 3: Working with Associated Components on page 111.

3. In the *Signal Name* field, enter:

vcc

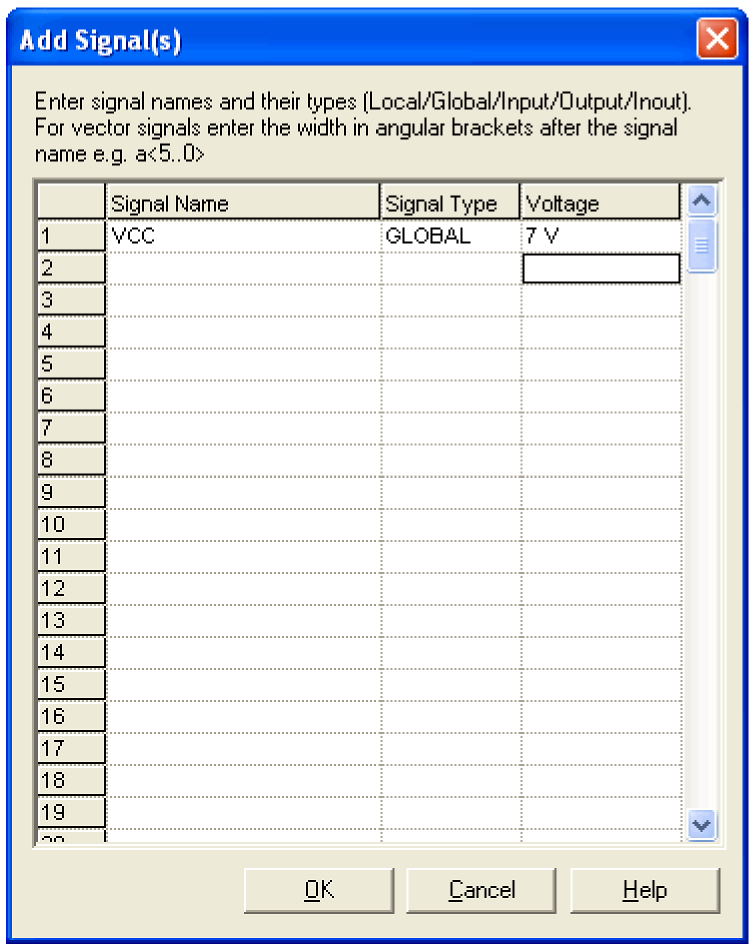
4. In the *Signal Type* field, select the scope of the `VCC` signal as *Global*.

Global signals are used to make sure that voltage signals with the same name in different blocks in a hierarchical design are connected together.

5. In the *Voltage* field, enter the voltage for the `VCC` signal as:

   `7`

   This completes the definition for the `VCC` signal.

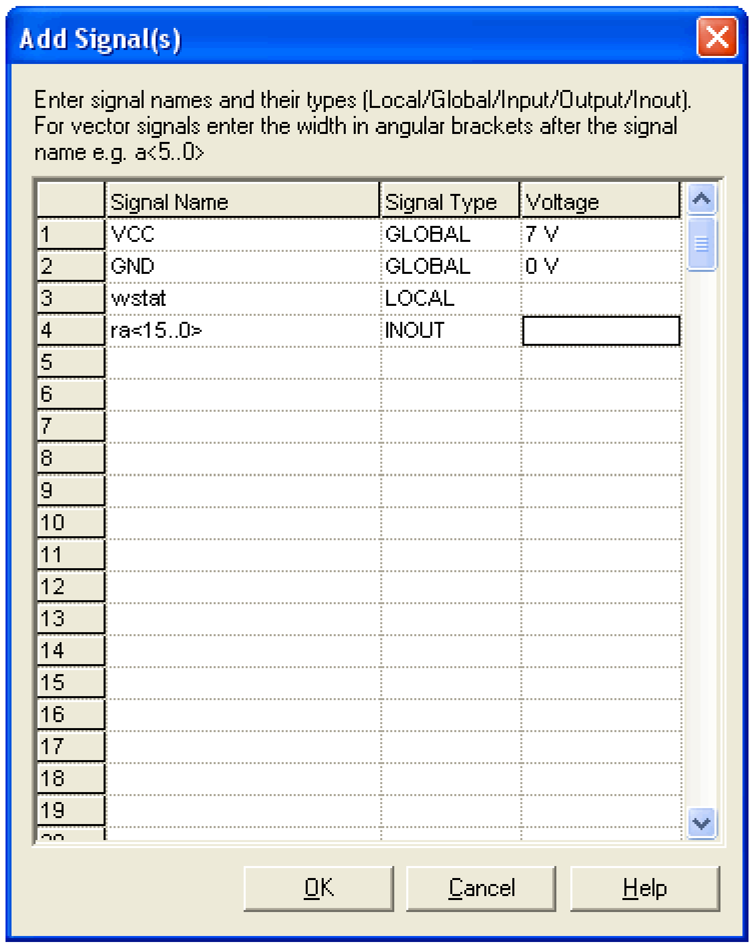


Note: The default unit for voltage is Volts. If you want to specify the voltage in millivolts, say, `75 Mv`, enter the voltage as:

`0.075`

6. Define the following signals:

Signal Name	**Signal Type**	**Voltage**
GND	GLOBAL	0
wstat	LOCAL	
ra<15..0>	INOUT	

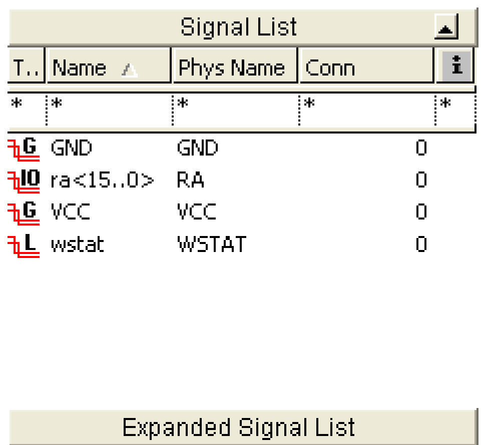


The signal `wstat` is a `LOCAL` signal. A local signal is a signal that is unique to a design. Local signals that have the same name in different designs will not be connected.

The signal `ra<15..0>` is a vectored signal. A vectored signal name indicates multiple bits on the same signal. You must separate the vector or bit range with two periods (..) and enclose it in angle brackets (< >). The bit order is <*MSB..LSB*> where `MSB` is the most significant bit and `LSB` is the least significant bit of the signal.

7. Click *OK* to add the signals in the design.

8. The signals are displayed in the Signal List. The icons in the *Type Icon* column in the Signal List indicate the scope of the signals.



For example, the icon next to the `VCC` signal indicates that `VCC` is a global signal. The next to the `ra<15..0>` signal indicates that `ra<15..0>` is an `INOUT` signal.

The Signal List lets you work with the signals in the design. It displays the list of signals in the design. Each row in the Signal List corresponds to a signal.

When you add a signal in the design, Design Editor automatically assigns a physical net name for the signal.

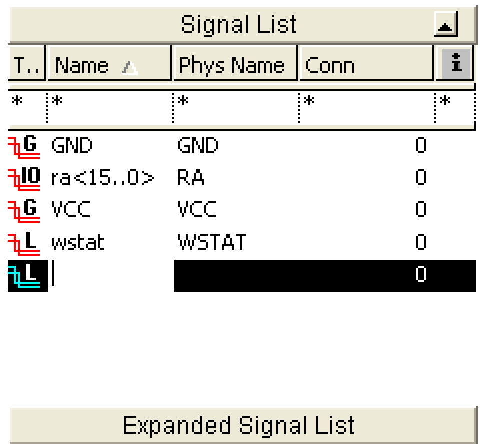
The *Conn* column in the Signal List displays the total number of component pins connected to each signal. The value `0` in the *Conn* column indicates that the signals are not connected to pins in the design. When you connect a signal to pins in the design, the value in the *Conn* column is automatically updated for the signal.

9. You can also add signals by pressing the *Insert* key in the Signal List.

a. Click in the Signal List.

b. Press the *Insert* key.

An empty row is added in the Signal List.

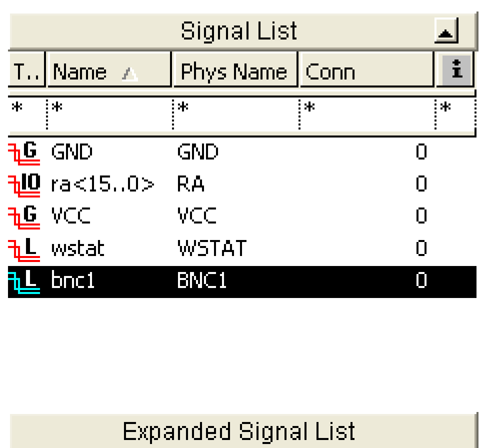


c. Enter the signal name as:

`bnc1`

d. Press *Enter*.

The `bnc1` signal is added as a local signal in the Signal List.



10. Another way to add signals in the design is to add pre-defined signals in the design.

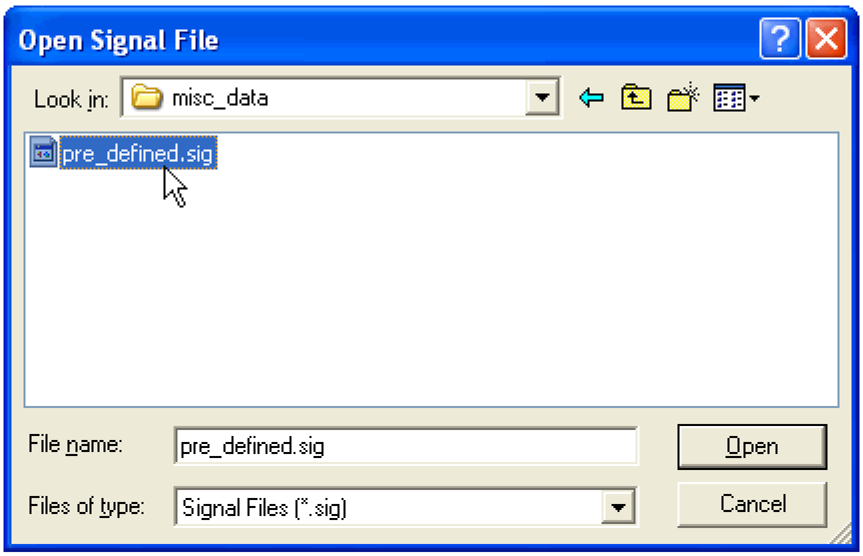
You can use the *Design – Save Signals To File* command to export the signals (and their voltage values, if any) in a design to a signal (`.sig`) file. You can then load the signal file into another design using the *Design – Load Predefined Signals* command. This lets you quickly add signals (along with their voltage values, if any) in other designs.

a. Choose *Design – Load Predefined Signals*.

The Open Signal File dialog box appears.

b. Select the `pre_defined.sig` located at:

`<your_work_area>\reference\misc_data`

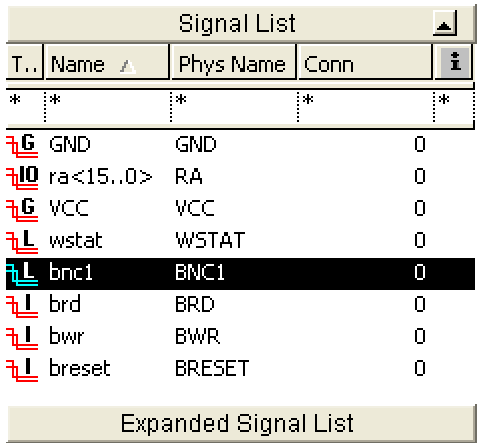


11. Click *Open*.

The following signals are automatically added in the Signal List.

```
brd
bwr
```

`breset`



12. You can modify the signal name, the physical net name, the scope of the signal, and its voltage in the Signal List. You will now modify the scope of the `ra<15..0>` signal from `INOUT` to `INPUT`.

   a. Select the `ra<15..0>` signal in the Signal List.

   b. Choose *Design – Change – Signal Scope – Input*.

   The scope of the `ra<15..0>` signal is displayed as  in the Type Icon column. This icon indicates an `INPUT` signal.

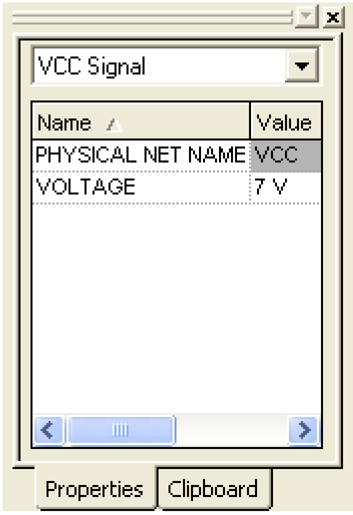
13. You will now change the voltage of the `VCC` signal from `7 V` to `5 V`.

   a. Select the `VCC` signal in the Signal List.

   b. Choose *View – Properties Window*.

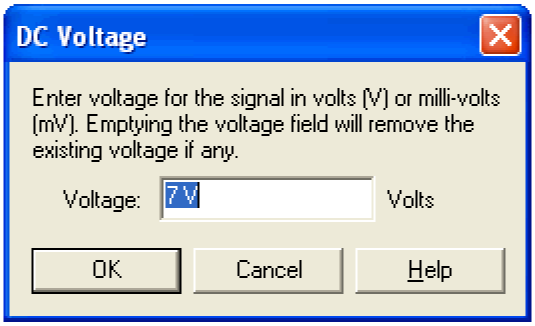
   The Properties window appears.

c. The Properties window displays the property `VOLTAGE` with the value 7 V.



d. Choose *Design – Change – DC Voltage*.

The DC Voltage dialog box appears.



e. Enter 5 in the Voltage field.

f. Click *OK*.

g. Click on the `VCC` signal in the Signal List.

The Properties window now displays the value of the `VOLTAGE` property as 5 V.

14. Choose *File – Save* to save the design.

Summary

You now know how to add signals in the design using the Add Signal(s) dialog box and the Signal List. You also learned the following:

- How to load a predefined list of signals from a file.
- The syntax to be used when adding vectored signals.
- Design Editor automatically assigns physical net names to the signals you add in the design.
- How to change the scope and voltage of signals.

For More Information

See:

Working with Components and Connectivity chapter of *Allegro Design Editor User Guide*.

Lesson 2-3: Adding Connectivity

Overview

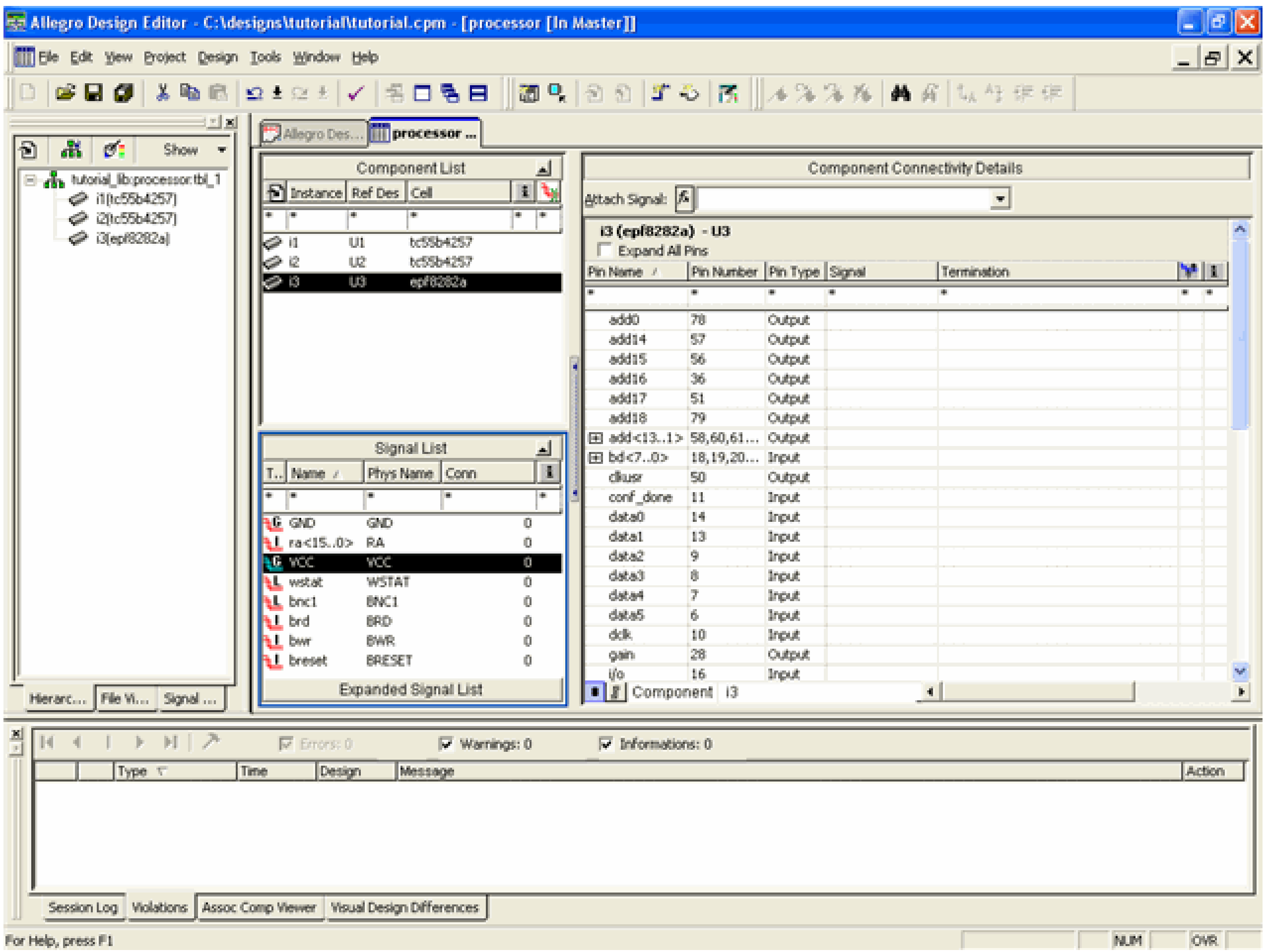
The previous lessons showed you how to add components and signals in the design. In this lesson, you will learn how to capture the connectivity information for the design by connecting component pins to signals.

The Component Connectivity Details pane (CCP) in Design Editor provides a spreadsheet-based interface to quickly add pin-signal connectivity, apply terminations, pullups and pulldowns, and assign signal integrity models to component pins.

We will discuss different ways to quickly interconnect the design.

Procedure

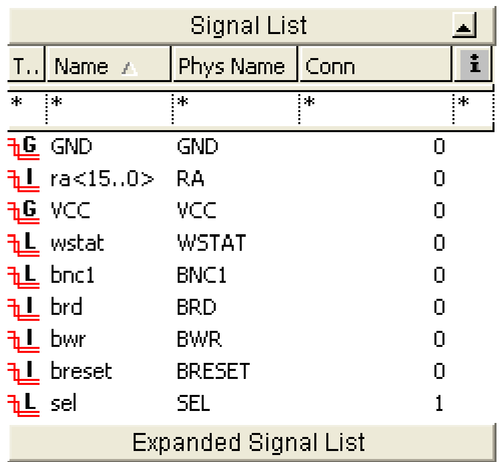
1. Double-click on the `epf8282a` component in the Component List to display the Component Connectivity Details pane.



2. In the *Signal* cell next to the pin `data1`, type the signal name:
   `sel`
3. Press *Enter* or click in another cell.

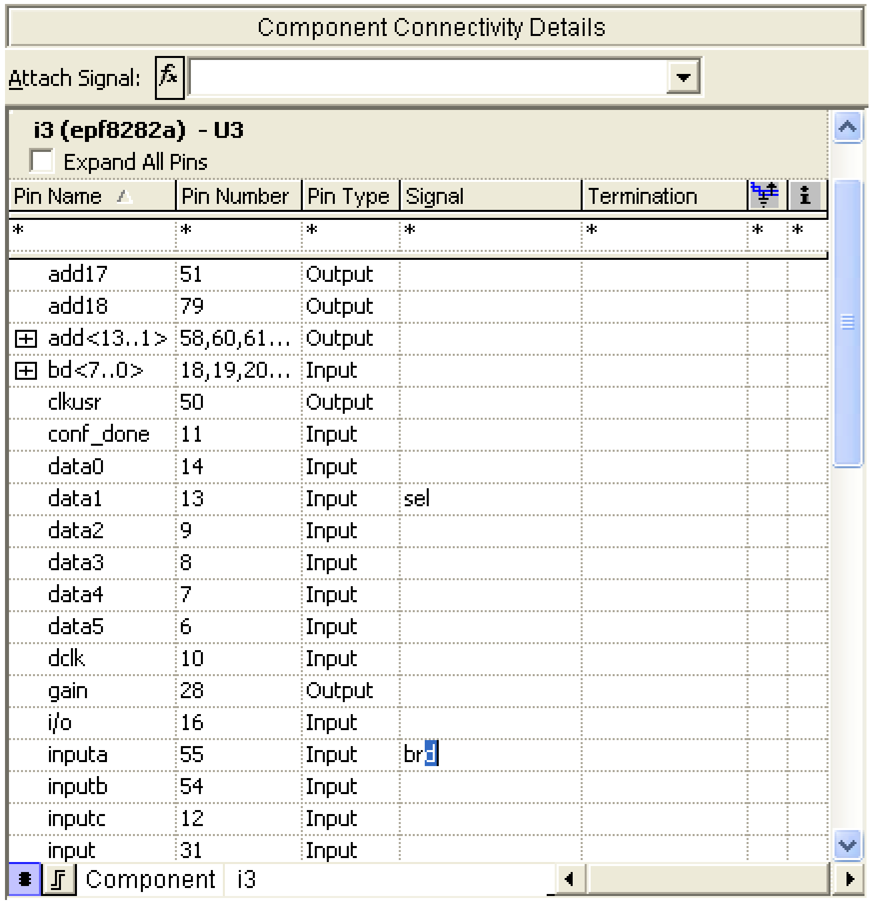
   The pin `data1` is connected to the signal `sel`.

   Note that the signal `sel` is automatically added in the Signal List because a signal with the same name does not exist in the design. The *Conn* column in the Signal List indicates that the signal `sel` has one connection in the design.



4. In the Signal cell next to the pin `inputa`, type:

`br`



The signal name auto completes to `brd`.

Design Editor auto completes signal names when you type them in the *Signal* column in the Component Connectivity Details pane.

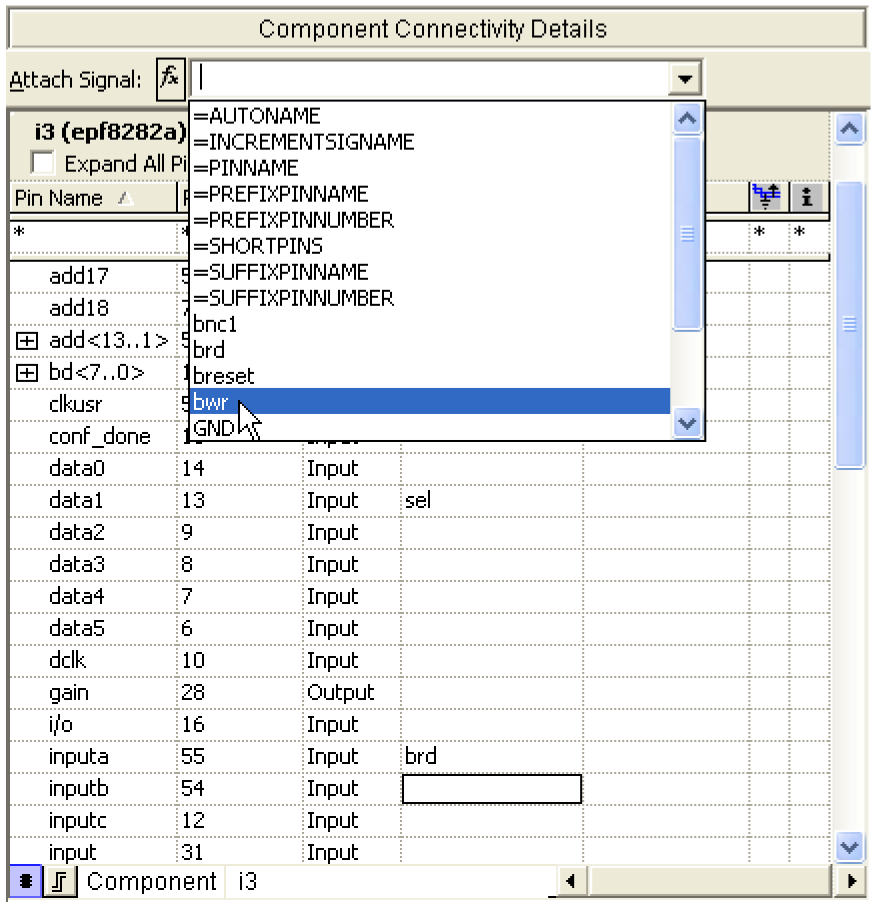
5. Press *Enter* or click in another cell.

The pin `inputa` is connected to the signal `brd`.

6. If you want to pick signal names from the Signal List in the design, you can select a pin in the Component connectivity Details pane or select the *Signal* cell next to a pin and then select a signal from the Attach Signal drop-down list.

a. Click in the *Signal* cell next to the pin `inputb`.

b. Click the *Attach Signal* drop-down list and choose the `bwr` signal.

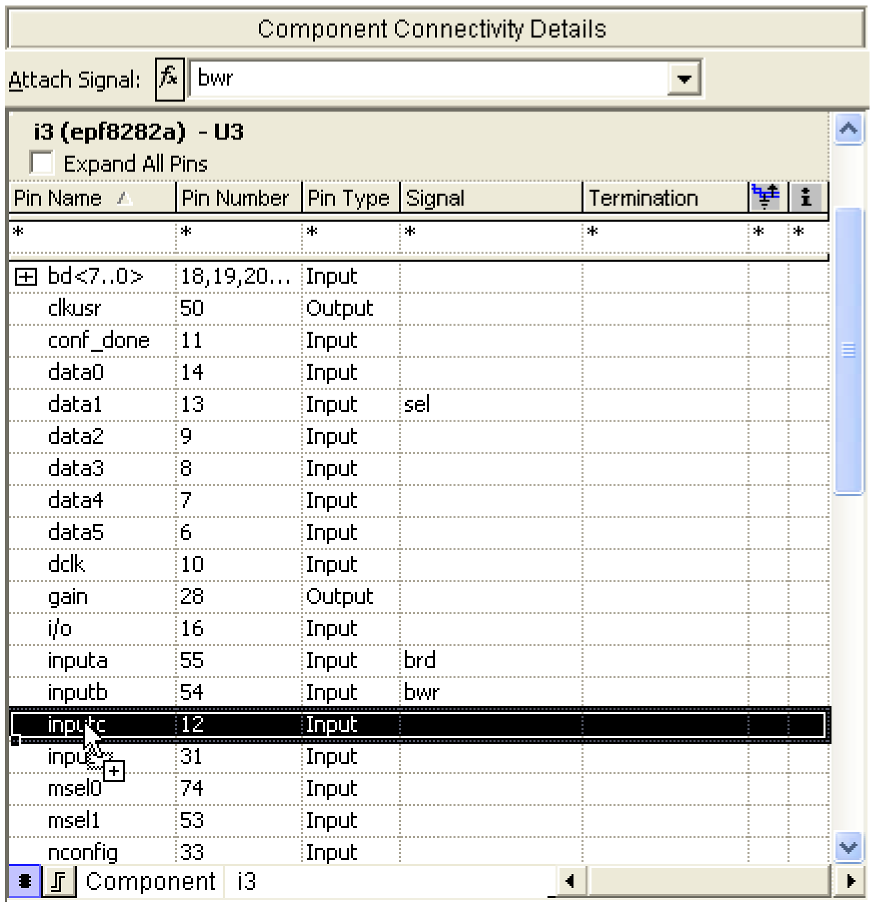


The pin `inputb` is connected to the `bwr` signal.

7. Another way to add connectivity is to select a signal in the Signal List and drag and drop the signal on a pin in the Component Connectivity Details pane.

a. Select the `breset` signal in the Signal List.

b. Keeping the left-mouse button pressed, drag and drop the signal on the pin `inputc` in the Component Connectivity Details pane.



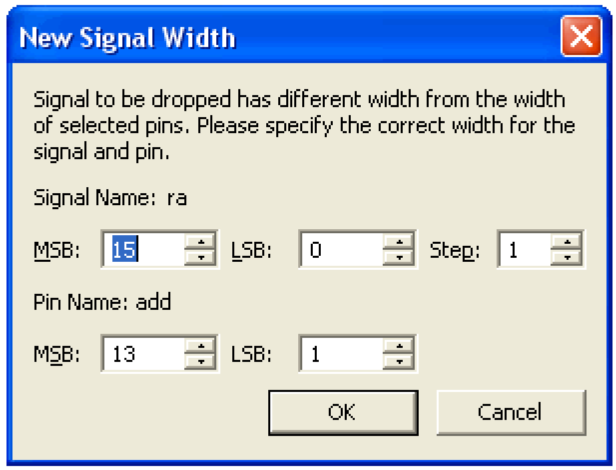
The pin `inputc` is connected to the `breset` signal.

If you drag and drop a vectored signal that has a different width from the size of the pin, the New Signal Width dialog box appears. Specify the bits of the signal you want to connect to the pin.

a. Select the `ra<15..0>` signal in the Signal List.

b. Keeping the left-mouse button pressed, drag and drop the signal on the vector pin `add<13..1>` in the Component Connectivity Details pane.

The New Signal Width dialog box appears.



c. Specify the MSB (most significant bit) for the `ra<15..0>` signal as `12` and the LSB (least significant bit) for the signal as `0`.

This indicates that the bits `0` to `12` of the `ra<15..0>` signal will be connected to the pin `add<13..1>`.

d. Click *OK* to connect the pin to the signal.

Note: You can click the ⊞ icon next to the pin `add<13..1>` or select the *Expand All Pins* check box to view the connectivity of each bit of the vector pin.

8. Another way to connect a pin to a signal is to enter the signal name in the *Signal* cell next to the pin.

Click in the *Signal* cell next to the pin `bd<7..0>` and type:

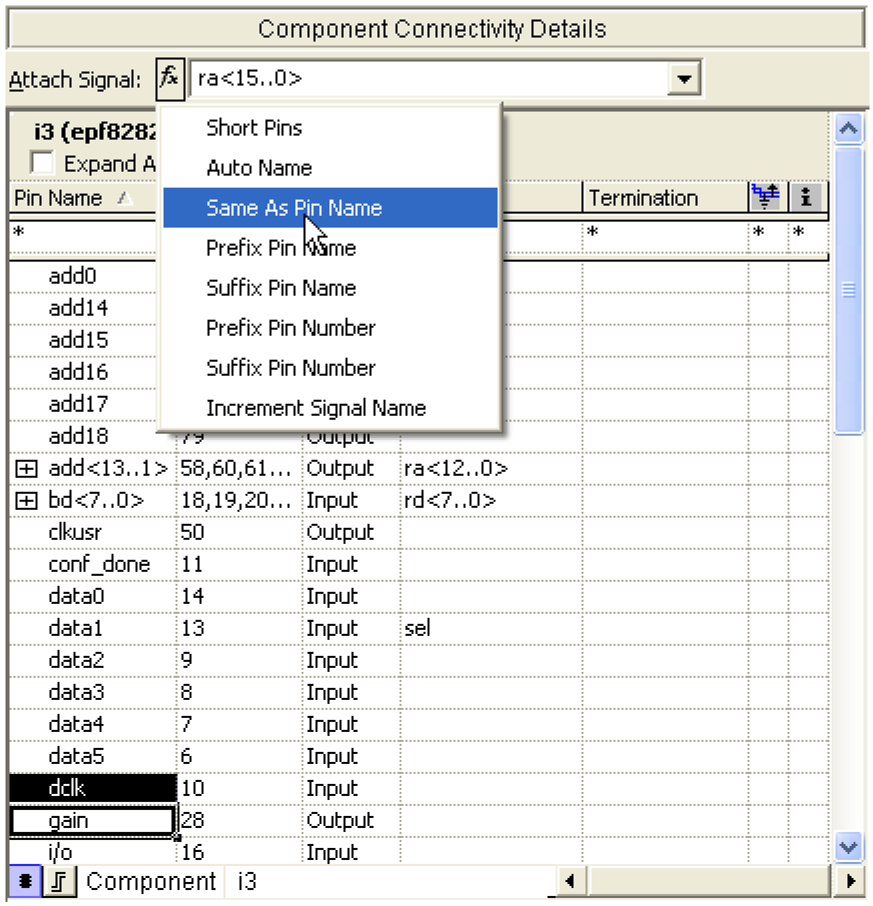
```
rd<7..0>
```

The pin `bd<7..0>` is connected to the signal `rd<7..0>`.

9. You can also use functions to automatically generate the signal names for pins.

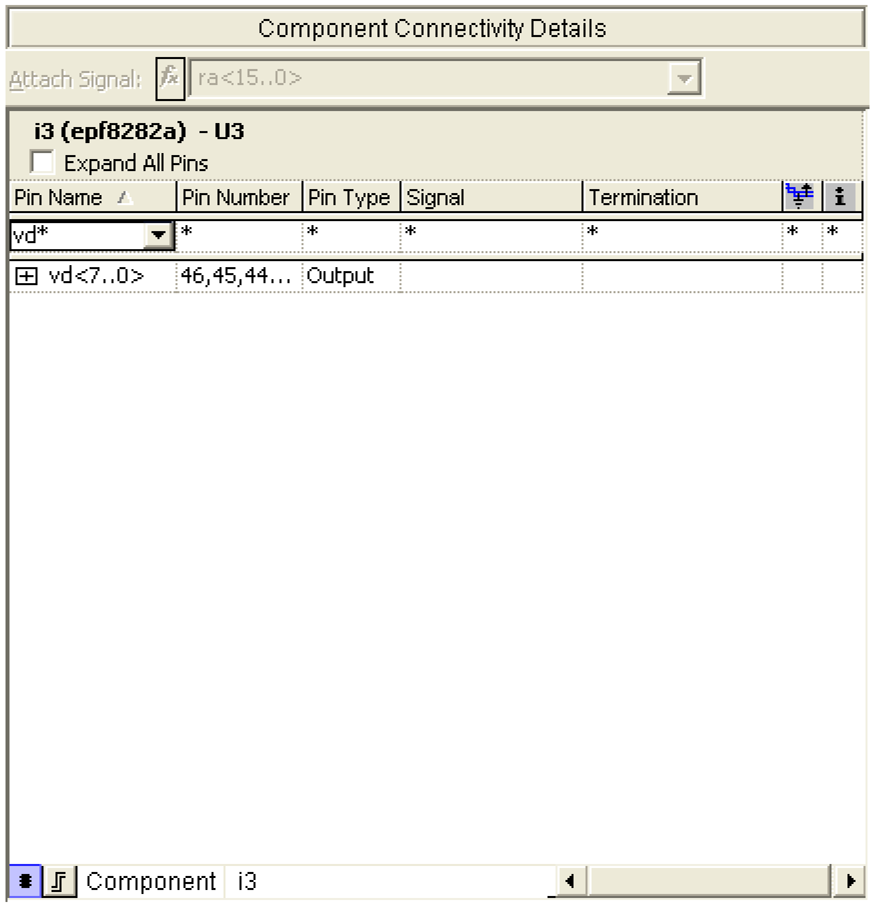
a. Select the pins `dclk` and `gain` or select the *Signal* cells next to the pins.

b. Click the button and choose *Same as Pin Name* to generate signal names that have the same name as the pin names.



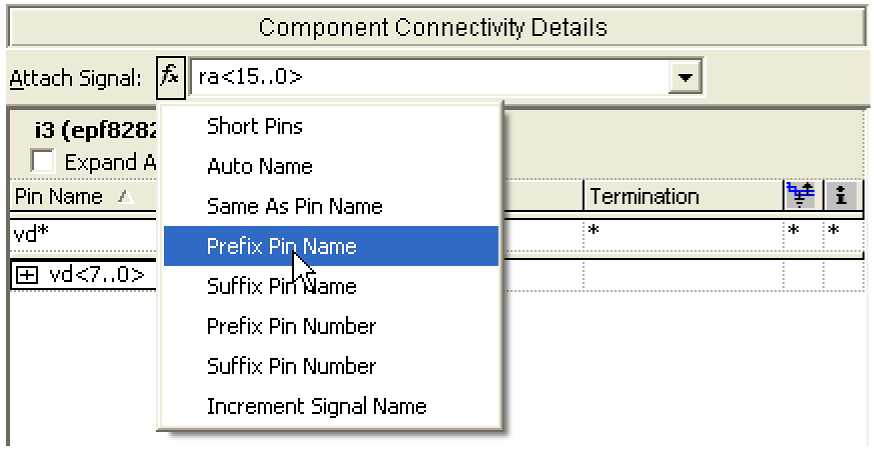
10. You can use filters to view only the pins on which you want to view or edit connectivity.

a. Type vd* in the *Pin Name* column filter and press *Enter* to view only the pin names starting with the name vd.

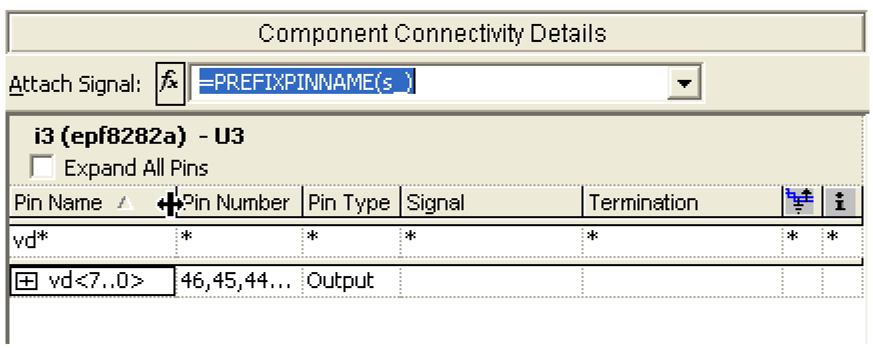


b. Select the pin `vd<7..0>`.

c. Click the button and choose *Prefix Pin Name*.



d. Type the prefix as s_ and press *Enter*.

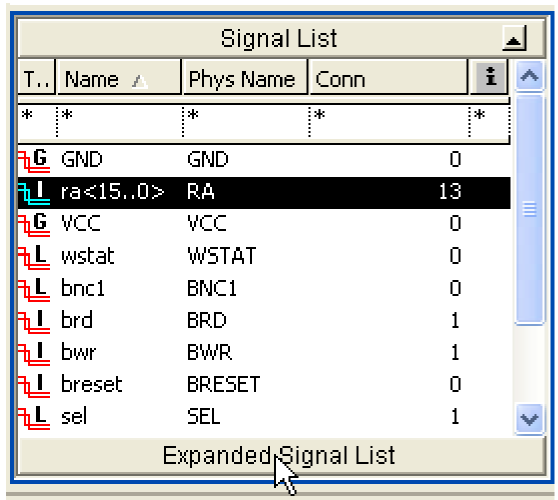


The pin `vd<7..0>` is connected to the `s_vd<7..0>` signal.

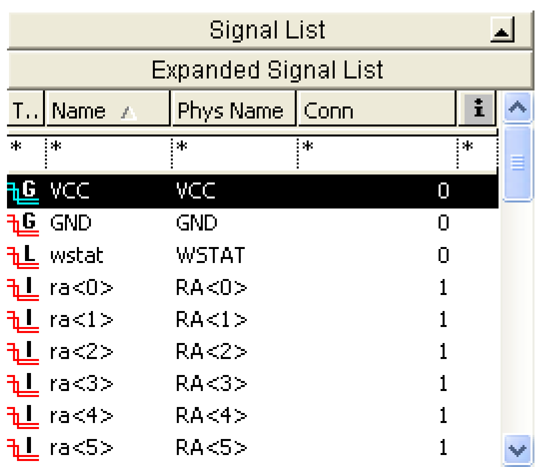
e. Type * in the *Pin Name* column filter and press *Enter* to view all the pins of the `epf8282a` component.

11. By default, vectored signals are displayed in the Signal List. Displaying the bits of a vectored signal in the Signal List lets you quickly connect bits of the vectored signal to component pins.

a. Click the Expanded Signal List button in the Signal List.



The bits of vectored signal `ra<15..0>` are displayed in the Signal List.

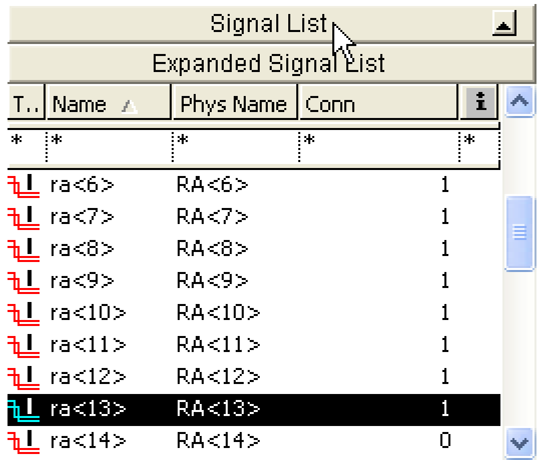


b. Select the bit `ra<13>` in the Expanded Signal List.

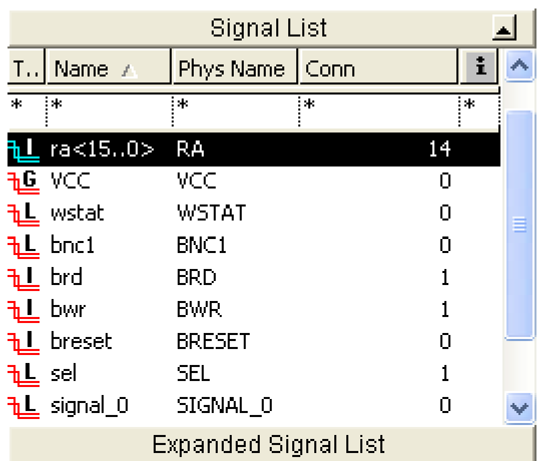
c. Keeping the left-mouse button pressed, drag and drop the signal on the pin `rdynbusy` in the Component Connectivity Details pane.

The pin `rdynbusy` is connected to the bit `ra<13>` of the vectored signal `ra<15..0>`.

d. Click the Signal List button.

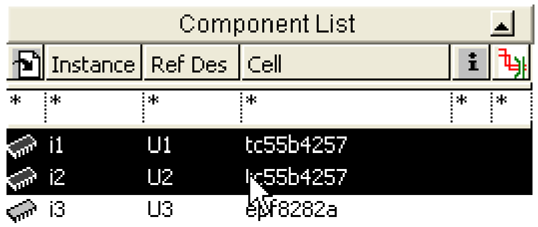


The vectored signals in the design are now displayed in non-expanded format.



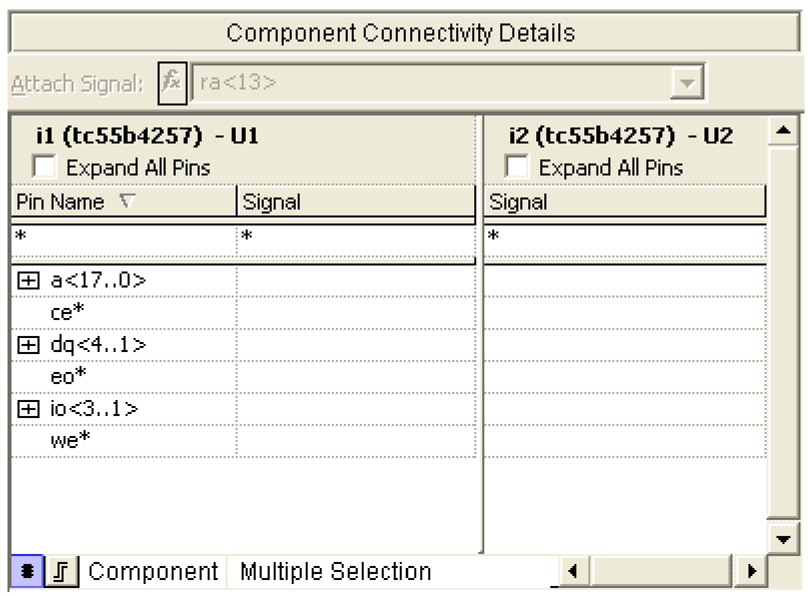
12. You can view and edit the connectivity for more than one component at the same time. This helps you quickly capture connectivity information on components that require similar connectivity.

a. Select the two instances of the `tc55b4257` component in the Component List.

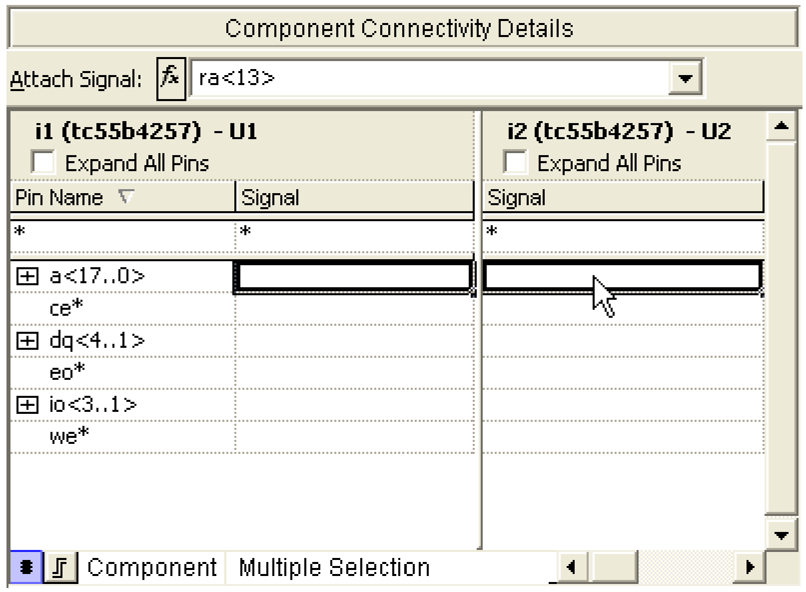


b. Choose *Design – Edit Connectivity*.

A multiselect Component Connectivity Details pane for both the instances with one Pin Name column and two Signal columns is displayed.



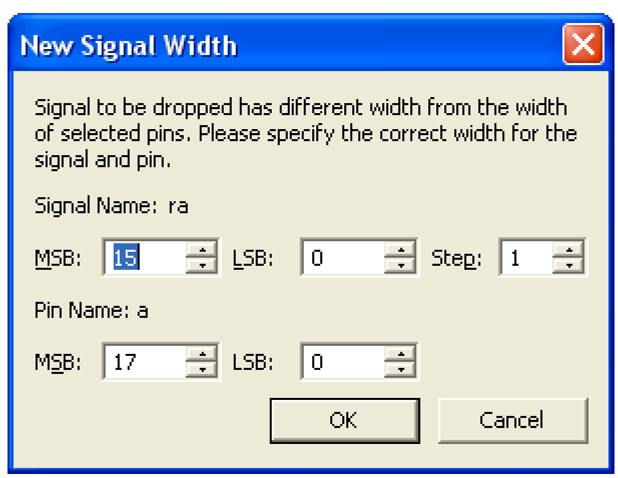
c. Press the *Ctrl* key and click in the *Signal* cells next to the pin a<17..0>.



d. Select the `ra<15..0>` signal in the Signal List.

e. Keeping the left-mouse button pressed, drag and drop the signal on the pin a<17..0> in the Component Connectivity Details pane.

The New Signal Width dialog box appears.



f. Specify the MSB (most significant bit) for the pin a<17..0> as 15.

This indicates that the `ra<15..0>` signal will be connected to the bits `0` to `15` of the pin `a<17..0>`.

g. Click *OK*.

The New Signal Width dialog box appears.

h. Specify the MSB (most significant bit) for the pin `a<17..0>` as `15`.

i. Click *OK*.

The pin `a<17..0>` on both the instances of the component are connected to the `ra<15..0>` signal.

j. Press the *Ctrl* key and click in the *Signal* cells next to the pin `dq<4..1>`.

k. Select the `rd<7..0>` signal in the Signal List.

l. Keeping the left-mouse button pressed, drag and drop the signal on the pin `dq<4..1>` in the Component Connectivity Details pane.

The New Signal Width dialog box appears.

m. Specify the MSB (most significant bit) for the `rd<7..0>` signal as `3`.

n. Click *OK*.

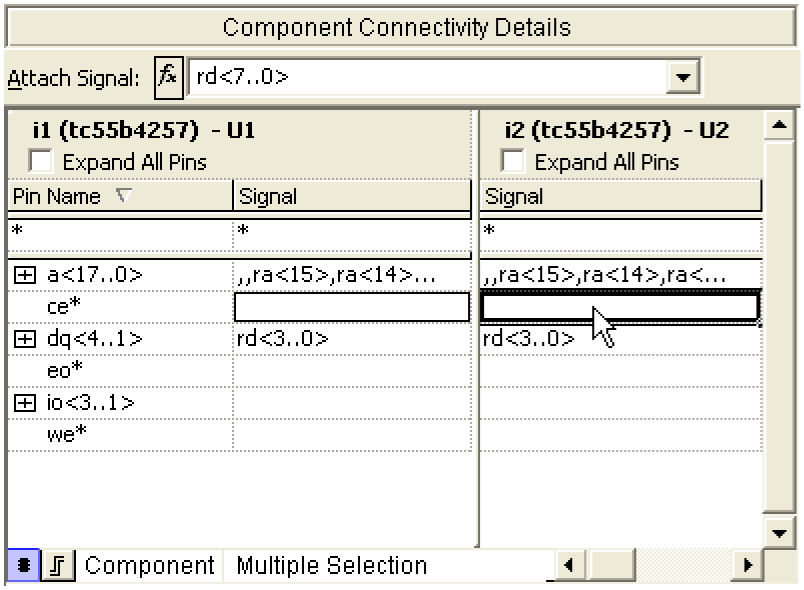
The New Signal Width dialog box appears.

o. Specify the MSB (most significant bit) for the `rd<7..0>` signal as `3`.

p. Click *OK*.

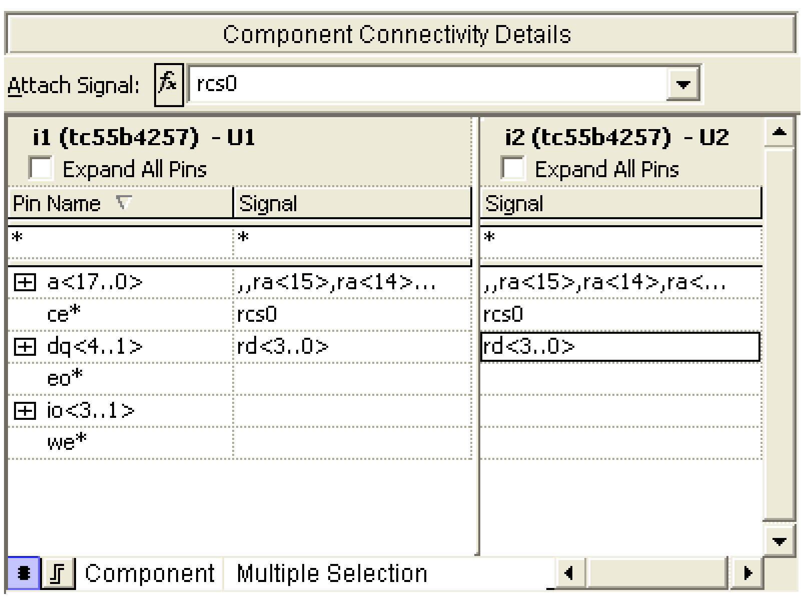
The pin `dq<4..1>` on both the instances of the component are connected to the `rd<3..0>` signal.

q. Press the *Ctrl* key and click in the *Signal* cells next to the pin `ce*`.



r. Type `rcs0` and press *Enter*.

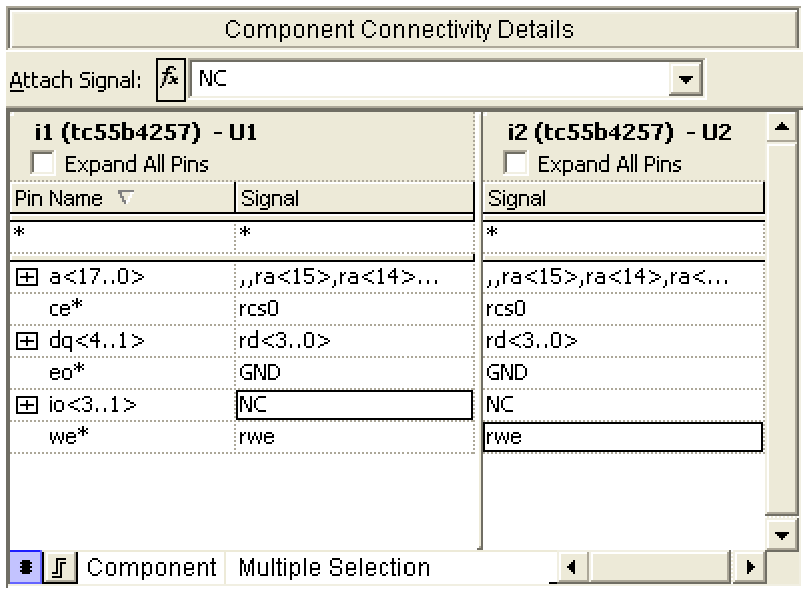
The pin `ce*` on both the instances of the component are connected to the `rcs0` signal.



s. Similarly connect:

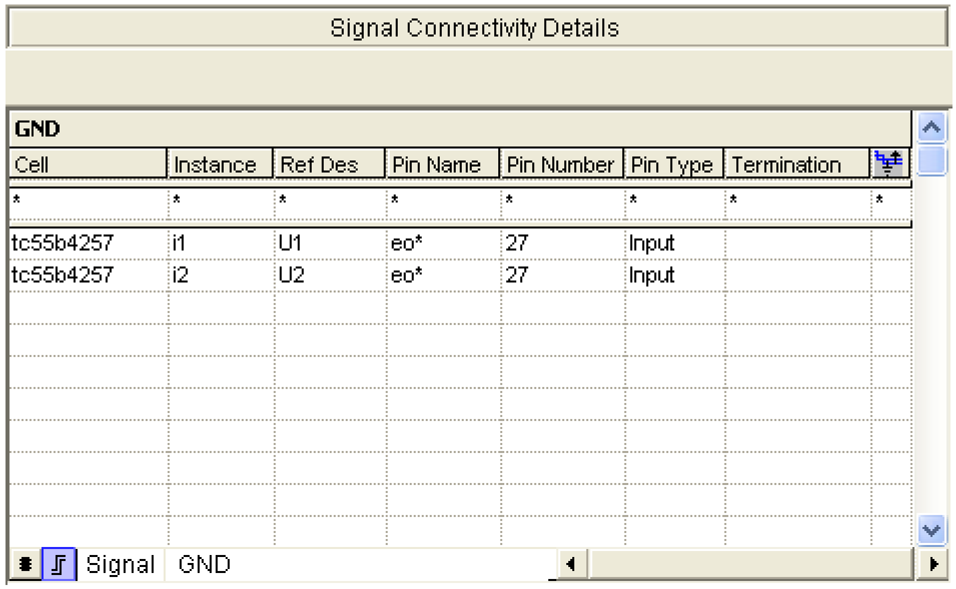
- Pin `eo*` to the `GND` signal.
- Pin `we*` to the `rwe` signal.
- Pin `io<3..1>` to the `NC` signal.

The `NC` signal is used to indicate a pin that is intentionally unused. The `NC` signal name transfers to the board layout in Allegro PCB Editor as a dummy net that will not be routed.



13. Double-click on the `GND` signal in the Signal List.

The connectivity information for the `GND` signal is displayed in the Signal Connectivity Details pane.

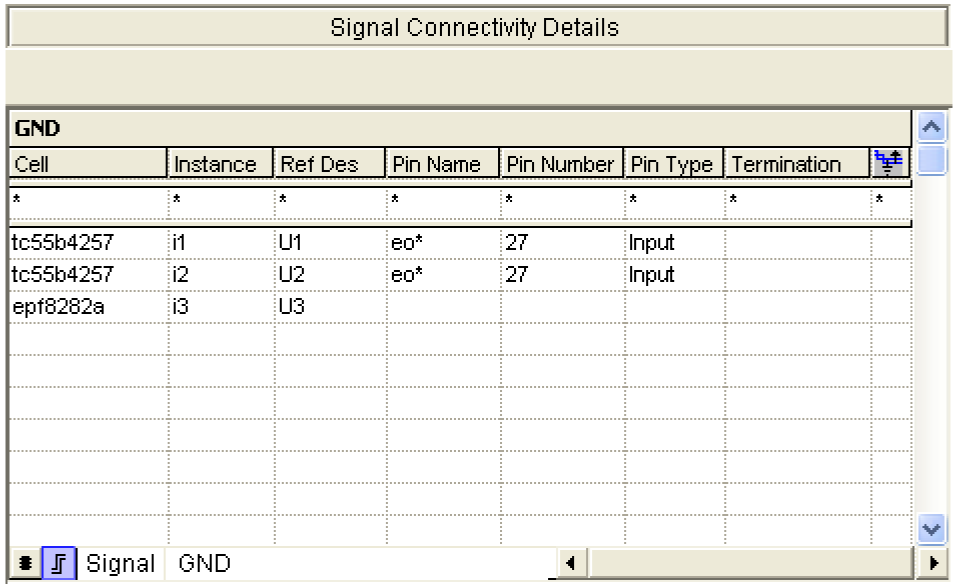


You can use the Signal Connectivity Details pane to quickly connect a signal to component pins.

a. In the Component List, select the `epf8282a` component.

b. Keeping the left-mouse button pressed, drag and drop the component in the Signal Connectivity Details pane.

A new row is added for the component.



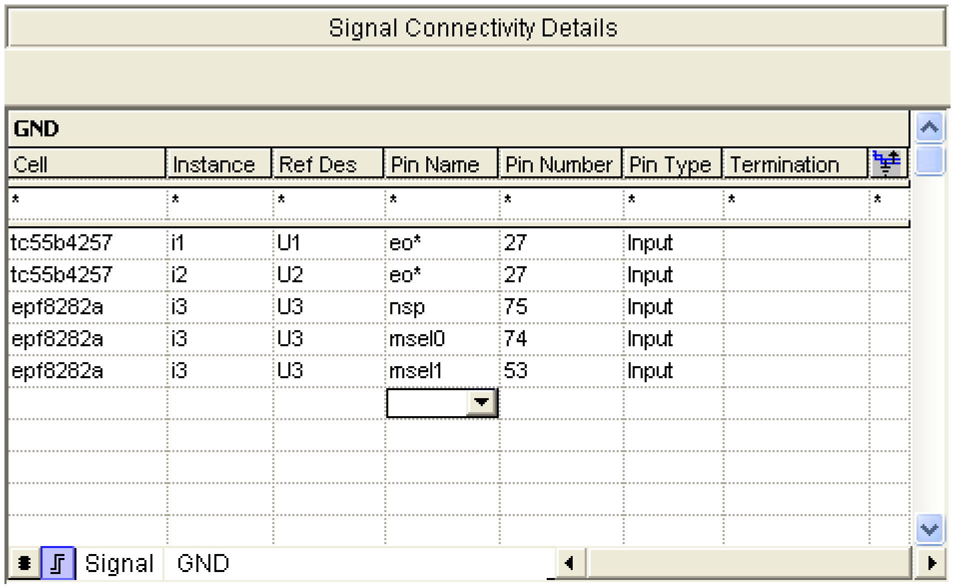
c. In the *Pin Name* column, click in the cell next to the `epf8282a` component.

d. Type `nsp` or click the drop-down list to select the pin `nsp`.

e. Click on any other cell in the Signal Connectivity Details pane.

The pin `nsp` is connected to the `GND` signal.

f. Repeat steps a to d described above to connect the `msel0` and `msel1` pins of the `epf8282a` component to the `GND` signal.



Signal Connectivity Details

GND

Cell	Instance	Ref Des	Pin Name	Pin Number	Pin Type	Termination	
*	*	*	*	*	*	*	*
tc55b4257	i1	U1	eo*	27	Input		
tc55b4257	i2	U2	eo*	27	Input		
epf8282a	i3	U3	nsp	75	Input		
epf8282a	i3	U3	msel0	74	Input		
epf8282a	i3	U3	msel1	53	Input		

Signal GND

14. When you copy and paste a component in the Component List, its connectivity and property information are also copied. This feature lets you capture connectivity and property information on one instance of a component and quickly replicate the connectivity and property information on additional instances of the same component you want to use in your design.

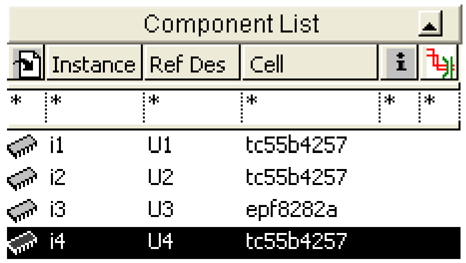
a. Double-click on the `tc55b4257` component with the instance name `i1` in the Component List.

The connectivity for the component is displayed in the Component Connectivity Details pane.

b. Choose *Edit – Copy* or press *Ctrl + C.*

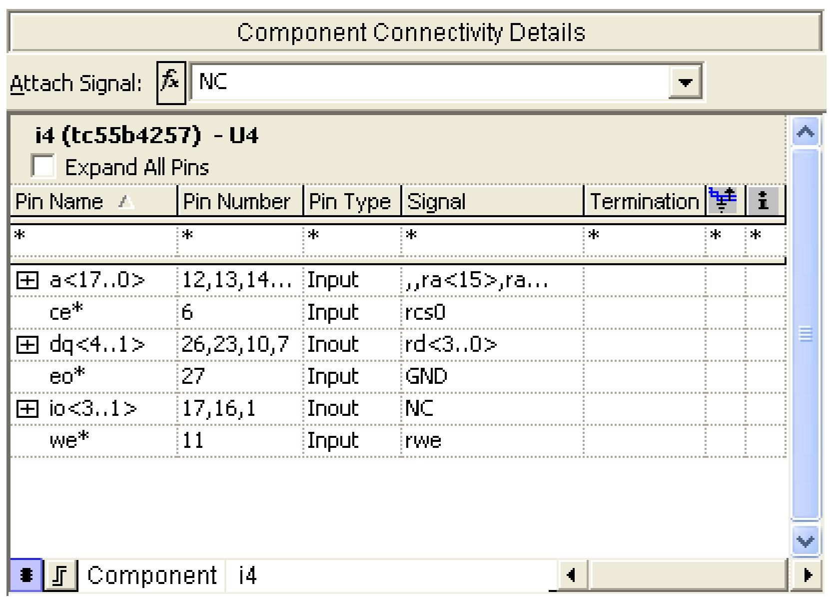
c. Choose *Edit – Paste* or press *Ctrl + V.*

A new instance of the `tc55b4257` component with the instance name `i4` is added in the Component List.



d. Select instance `i4` of the `tc55b4257` component in the Component List.

The connectivity for the component is displayed in the Component Connectivity Details pane.



Note that the connectivity information on the component has also been copied.

e. Connect the pin `ce*` to the `rcs1` signal.

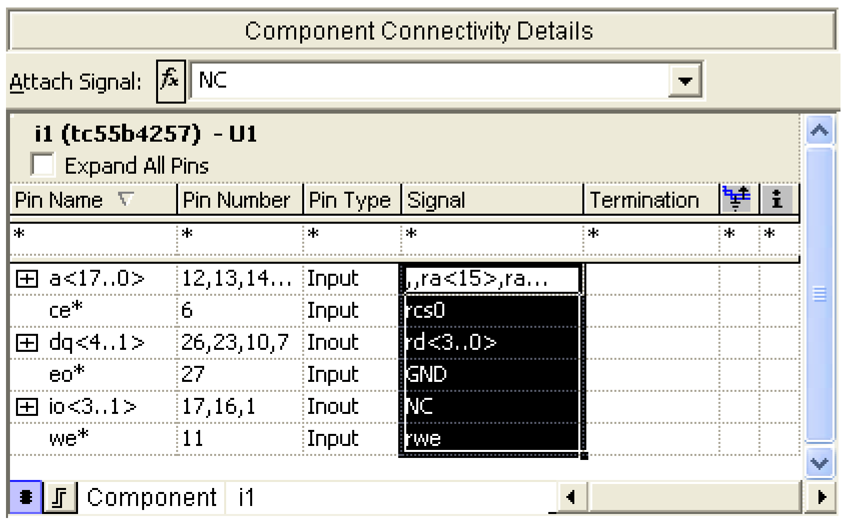
15. The paste special feature in Design Editor lets you copy connectivity from one component and paste on another component after making the required changes.

a. Add an instance of the `tc55b4257` component in the design.

For more information on adding components in the design, see Lesson 2-1: Adding Components on page 56.

b. Select instance `i1` of the `tc55b4257` component in the Component List.

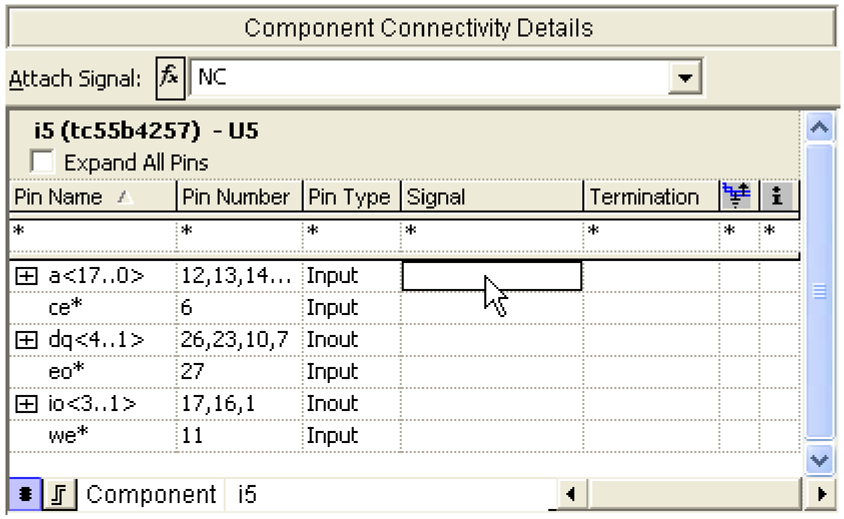
c. In the Component Connectivity Details pane, select all the signals in the *Signal* column.



d. Choose *Edit – Copy* or press *Ctrl + C* to copy the signal names.

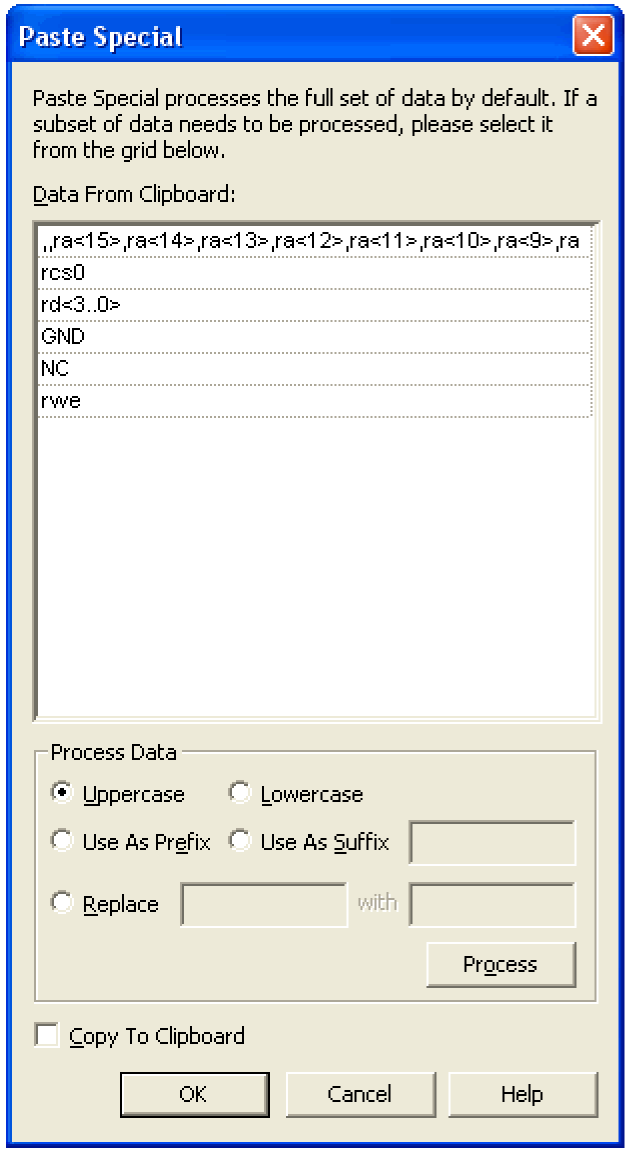
e. Select instance `i5` (the instance you added in step a above) of the `tc55b4257` component.

f. Click in the first cell in the *Signal* column in the Component Connectivity Details pane.



g. Choose *Edit – Paste Special*.

The Paste Special dialog box appears.

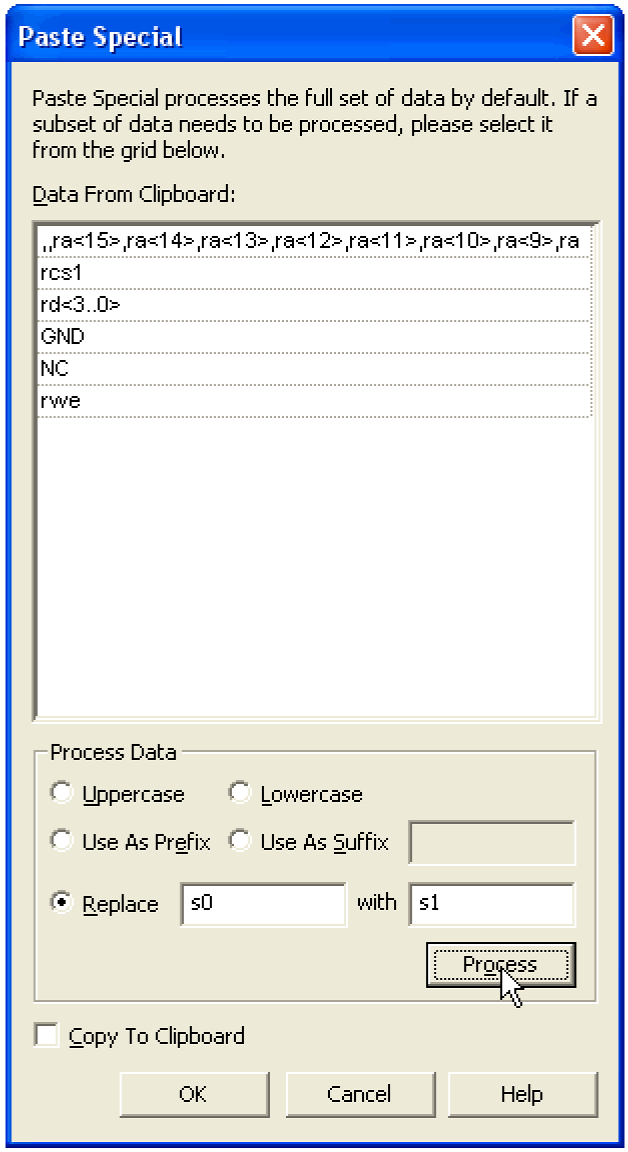


Here you will replace the `rcs0` signal with `rcs1`.

h. Select the *Replace* option.

i. Type `s0` in the first field and `s1` in the second field.

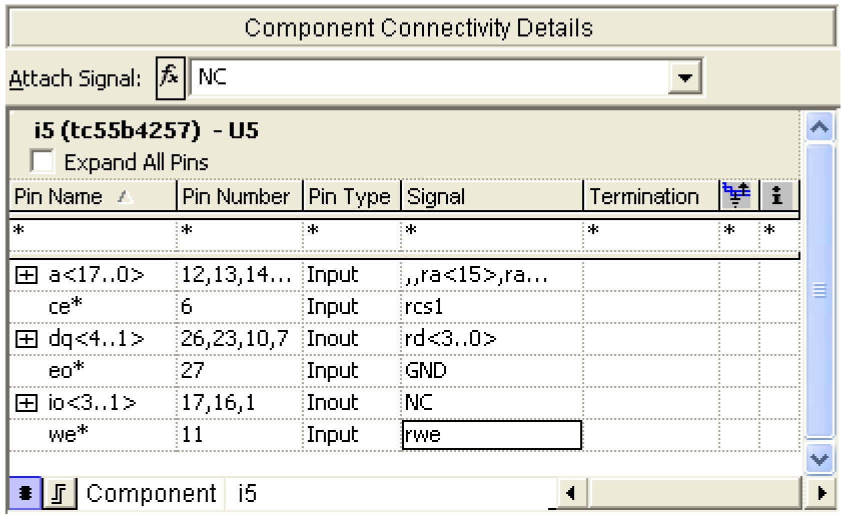
j. Click the *Process* button.



k. The signal `rcs0` is changed to `rcs1`.

l. Click *OK*.

The modified connectivity information is pasted in the *Signal* column of the Component Connectivity Details pane.



16. Choose *File – Save* to save the design.

Tip

You can also copy signal names from the Component Connectivity Details pane or from other applications such as Microsoft Excel and paste it in the *Signal* column in the Component Connectivity Details pane.

Summary

You now know how to use the Component Connectivity Details pane and the Signal Connectivity Details pane in Design Editor to quickly add connectivity information in the design. You also learned how to copy and paste connectivity information.

For More Information

See:

Working with Components and Connectivity chapter of *Allegro Design Editor User Guide*.

Lesson 2-4: Adding Comments in the Design

Overview

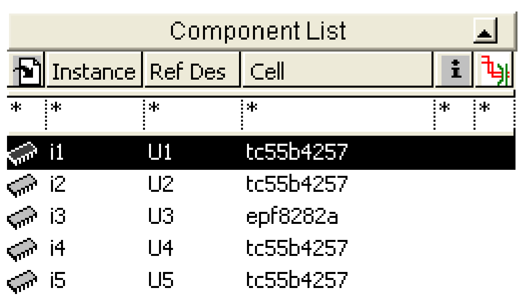
Design Editor allows you to add comments on components, signals and pins in the design. You can add comments on components in the Component List, on signals in the Signal List, and on pins in Component Connectivity Details pane.

The comments you add in the design can be displayed in reports and in the schematic generated for the design.

In this lesson, you will learn to add comments on components, signals and pins in the design.

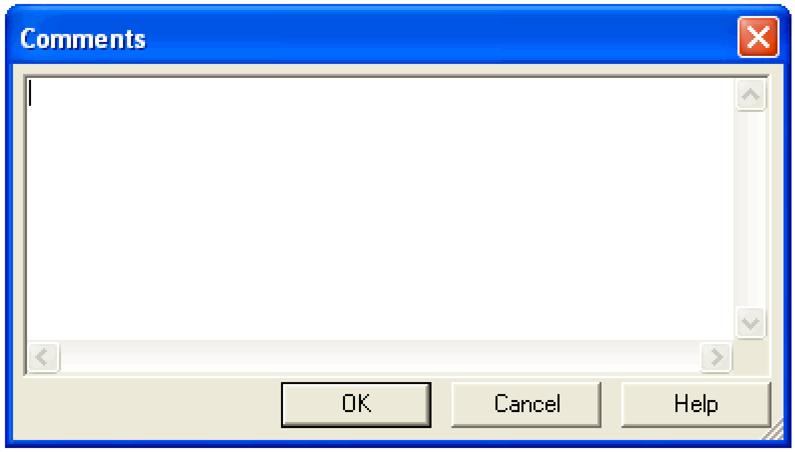
Procedure

1. In the Component List, select instance `i1` of the `tc55b4257` component.



2. Choose *Design – Comments – Insert Comment*.

The Comments dialog box appears.



3. Enter the comment:

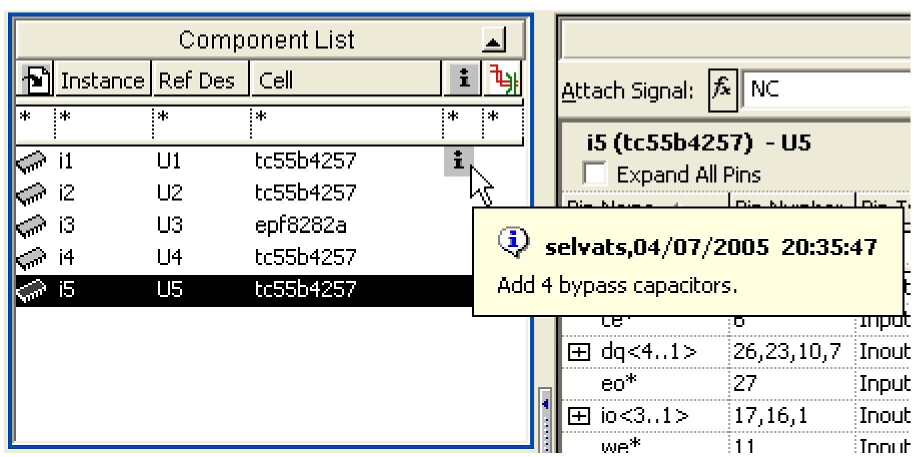
```
Add 4 bypass capacitors.
```

4. Click *OK*.

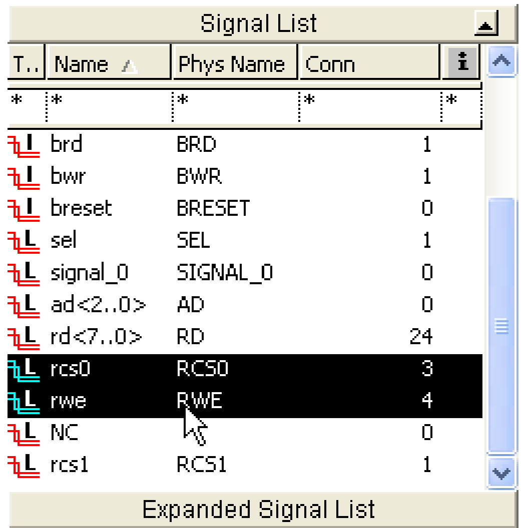
   The icon in the Comments column (the column with the icon) next to the component indicates that comments have been added on the component.

5. Place the mouse pointer over the icon next to the component.

   A tooltip displays the comment added on the component.



6. Select the signals `rcs0` and `rwe` in the Signal List.



7. Click the right-mouse button and choose *Comments – Insert Comments*.

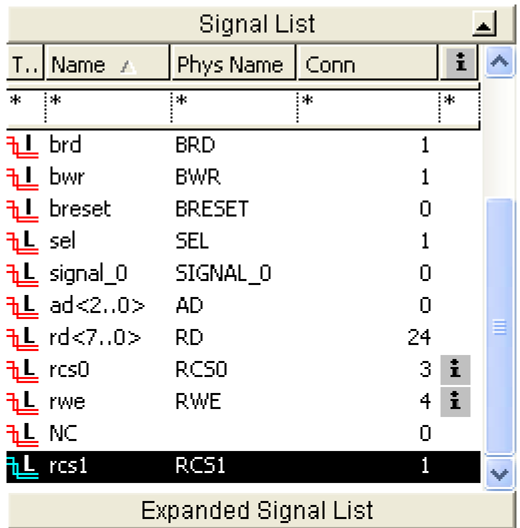
   The Comments dialog box appears.

8. Enter the comment:

   `Pullup this signal.`

9. Click *OK*.

The icon is displayed next to the signals in the Comments column (the column with the icon) in the Signal List.



10. In the Component Connectivity Details pane, double-click in the Comments column (the column with the icon) next to the pin `ce*`.

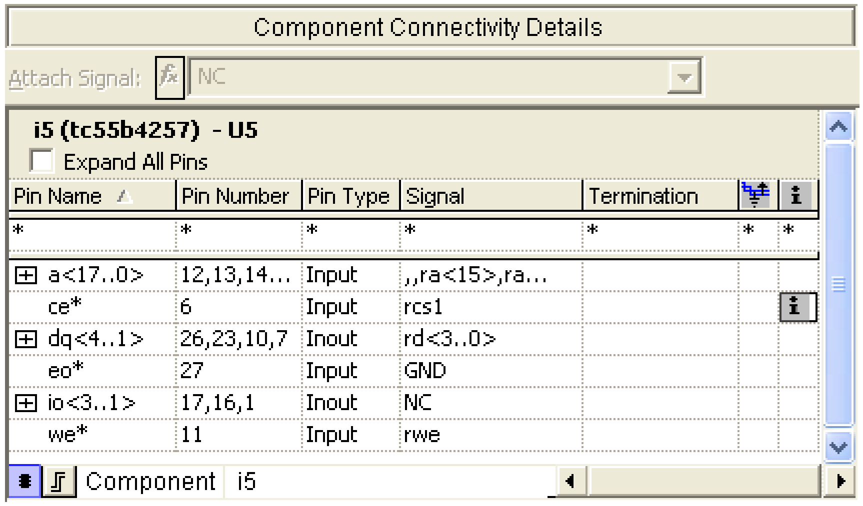
The Comments dialog box appears.

11. Enter the comment:

`Apply shunt termination.`

12. Click *OK*.

The icon is displayed next to the pin `ce*` in the Comments column in the Component Connectivity Details pane.



13. Choose *File – Save* to save the design.

Summary

You now know how to add comments on components, signals and pins in the design. You also learned how to view the comments on objects in the design.

For More Information

See:

<ins>Working with Components and Connectivity</ins> chapter of *Allegro Design Editor User Guide*.

Exercise

Click on the `epf8282a` component in the Component List and capture the pin-signal connectivity on the component as shown below.

Tip

Use the Signal List, Expanded Signal List, Component Connectivity Details pane and the Signal Connectivity Details pane to capture the connectivity information.

i3 (epf8282a) - U3

☐ Expand All Pins

Pin Name	Pin Num...	Pin Type	Signal	Termination
*	*	*	*	*
add0	78	Output	ra<14>	
add14	57	Output	rwe	
add15	56	Output	wstat	
add16	36	Output		
add17	51	Output	rcs3	
add18	79	Output	ra<15>	
⊞ add<13..1>	58,60,6...	Output	ra<12..0>	
⊞ bd<7..0>	18,19,2...	Input	rd<7..0>	
clkusr	50	Output	rcs2	
conf_done	11	Input	ncs	
data0	14	Input	data	
data1	13	Input	sel	
data2	9	Input	ba<3>	
data3	8	Input	ba<2>	
data4	7	Input	ba<1>	
data5	6	Input	ba<0>	
dclk	10	Input	dclk	
gain	28	Output	gain	
i/o	16	Input		
inputa	55	Input	brd	
inputb	54	Input	bwr	
inputc	12	Input	breset	
input	31	Input	mclk	
msel0	74	Input	GND	
msel1	53	Input	GND	
nconfig	33	Input	wait	
ncs	29	Input	hs	
nrs	48	Output	rcs0	
nsp	75	Input	GND	
nstatus	32	Input	oe	
ntrst	52	Input		
nws	30	Input	fpga	
⊞ rd<7..0>	4,3,2,1,...	Output	rd<7..0>	
rdclk	49	Output	rcs1	
rdynbusy	77	Output	ra<13>	
reset	15	Input	reset	
tck	72	Input		
tdi	73	Input		
tdo	27	Input		
vclka	34	Output	vclka	
vclkb	35	Output		
vclkc	37	Output	vclkc	
⊞ vd<7..0>	46,45,4...	Output	s_vd<7..0>	

Module 3: Working with Associated Components

Prerequisite

If you have not completed all the lessons in the previous modules, you must open the `tutorial.cpm` project located at `<your_work_area>\modules\assoc_comp\tutorial` in Design Editor and perform the steps described in this module.

For more information, see Understanding the Sample Design Files on page 14.

Lessons

This module consists of the following lessons:

Multimedia Demonstration

Click the link below to view a Flash-based multimedia demonstration of this module.

Working with Associated Components

Completion Time

1 hour for written lessons

11 minutes for multimedia demonstrations

Overview

Today's designs contain components, called associated components, that do not contribute to the logic of the design, but are a must for the correct functioning of the design. For example, bypass capacitors are needed for controlling power and ground bounce in the design. By definition, associated components are passive devices. Design Editor has classified associated components into three categories—terminations, bypass capacitors and pullup/pulldowns.

Traditional design entry tools do not capture the association between the parent object and the associated components. Wiring these components in the schematic is time consuming and error prone. Also, if you move or delete the parent object to which these associated components are attached in your schematic, you must ensure that the associated components are also moved or deleted.

The following lessons demonstrate how Design Editor allows you to quickly connect these devices to components and preserves their association with the components to which they are connected, making it easy to manage associated components in your design.

Lesson 3-1: Applying Terminations

Overview

Terminations are added to pins to ensure signal integrity. Terminations prevent the reflection of electrical signals occurring at the end of buses.

Design Editor supports applying standard terminations such as Series, Shunt, Thevenin and so on. The type of termination you can add on a pin depends on the pin type of the pin, which is selected for termination.

In this lesson, you will learn to apply shunt terminations in the design.

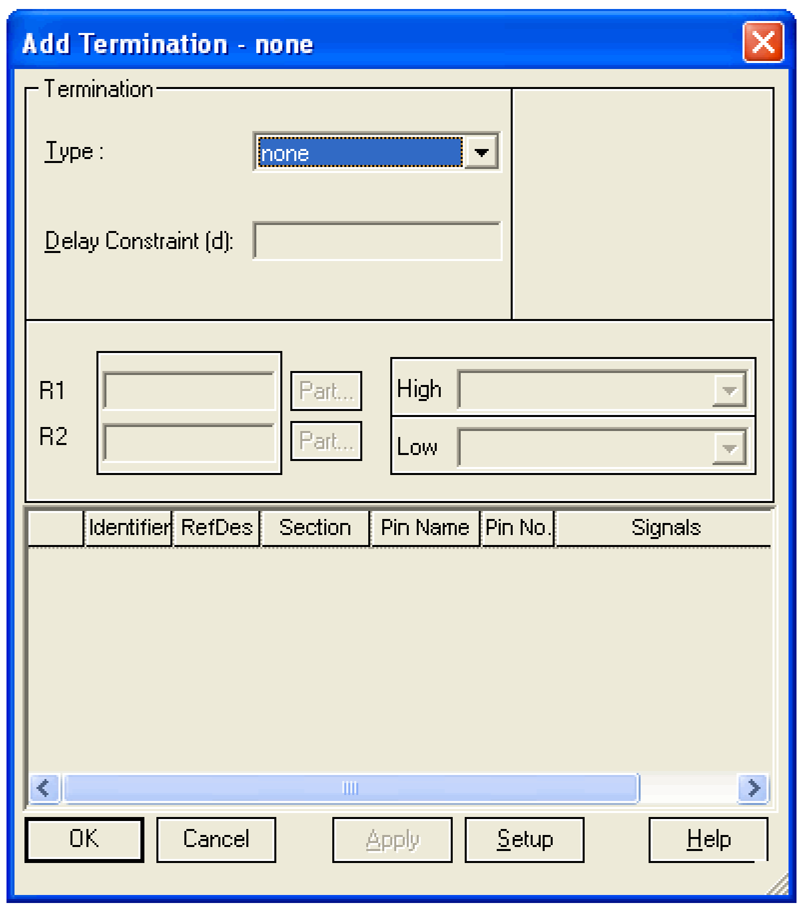
Procedure

1. In the Component List, select instance `i1` of the `tc55b4257` component.

   The connectivity information for the component is displayed in the Component Connectivity Details pane.

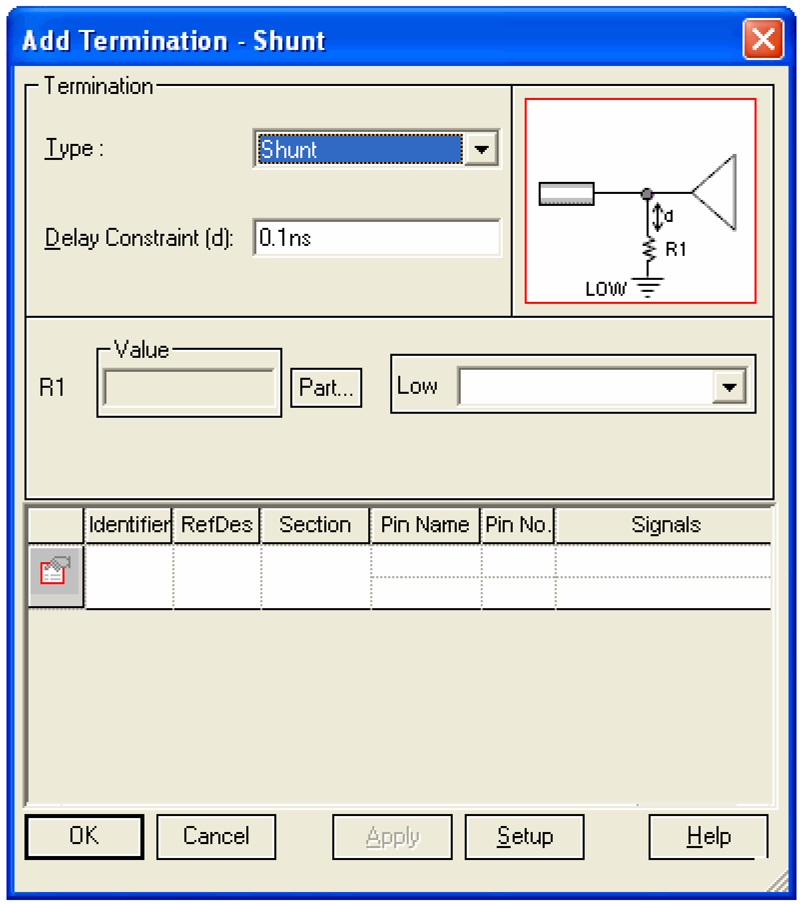
2. Double-click on the *Termination* column next to the pin `ce*`.

   The Add Termination dialog box appears.



3. From the *Type* drop-down list, choose *Shunt* to add a shunt termination.

   The Add Termination dialog box displays a graphical representation of the shunt termination scheme.



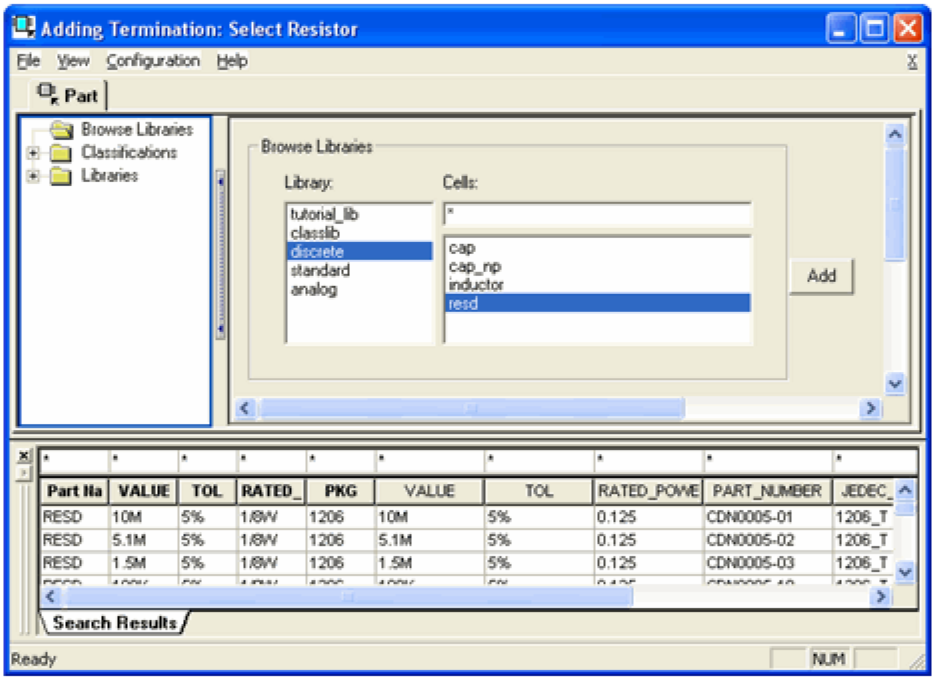
   The default delay constraint is `0.1ns` (nanoseconds). You can modify the delay value.

4. Click the *Part* button to select the resistor you want to use for the termination.

   The Component Browser appears.

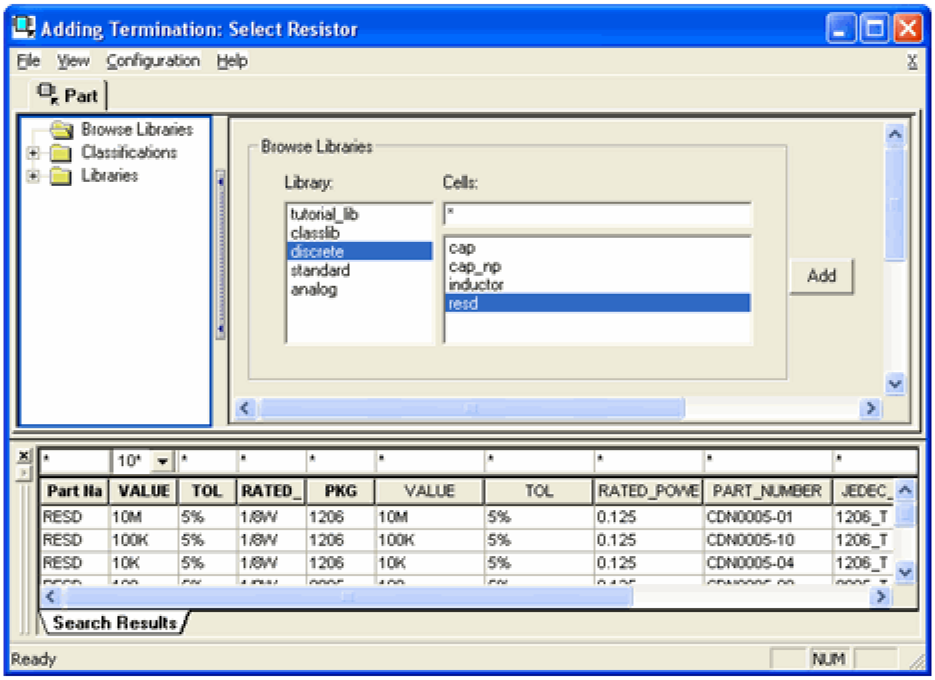
5. In the *Library* list, select the `discrete` library.

6. In the *Cells* list, select the `resd` component.



7. The part table rows for the resistor component are displayed in the *Search Results* pane.

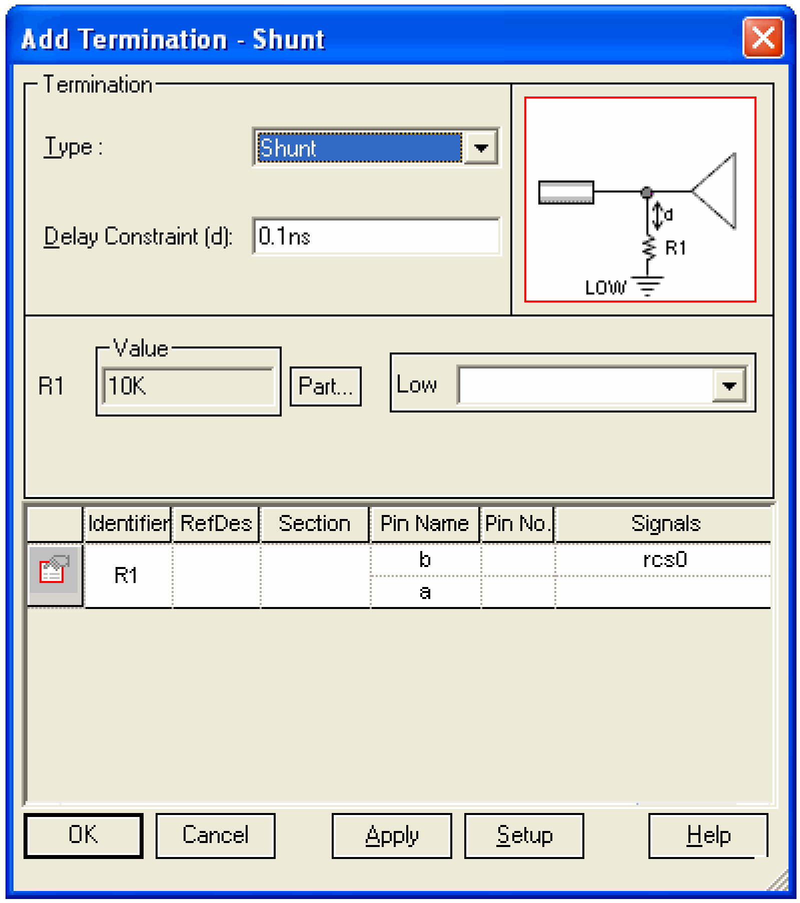
8. Type `10` in the filter above the VALUE field to display only the part table rows having the VALUE property with the value 10.



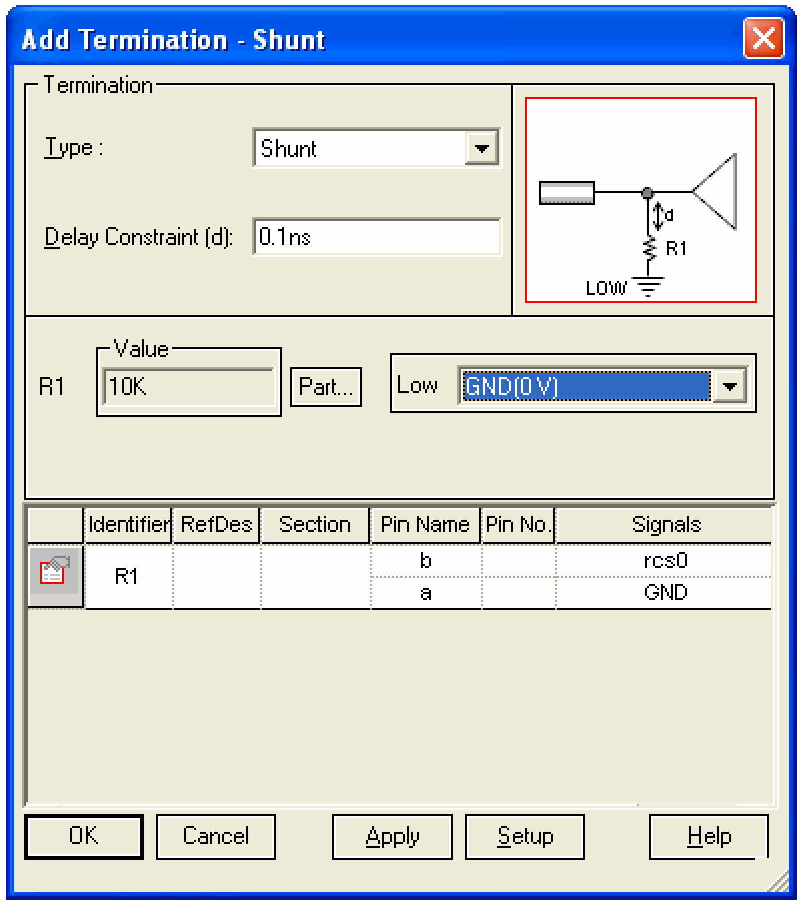
9. In the *Search Results* pane, select the physical part with the value `10K`.

10. Click the *Add* button.

The value of the resistor is displayed in the *Value* field in the Add Termination dialog box.

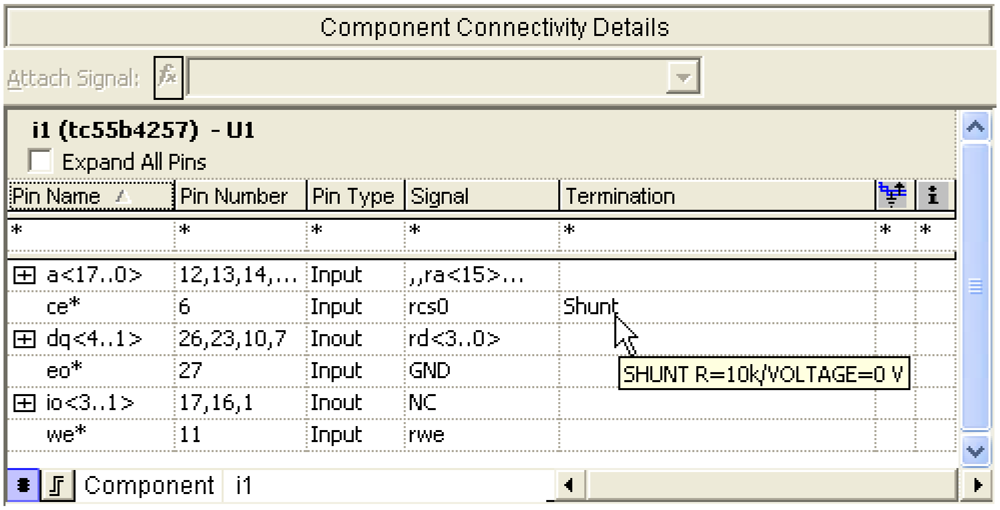


11. Click the *Low* drop-down list and select `GND` as the low signal.



12. Click *OK* to add the termination.

The termination type `Shunt` is displayed in the *Termination* column of the Component Connectivity Details pane.



If you place the mouse pointer over the termination, a tooltip displays the termination type, the resistor value and the voltage value of the signal.

You can copy the terminations added on a pin and paste it on another pin (on the same component or on another component) that supports the termination scheme.

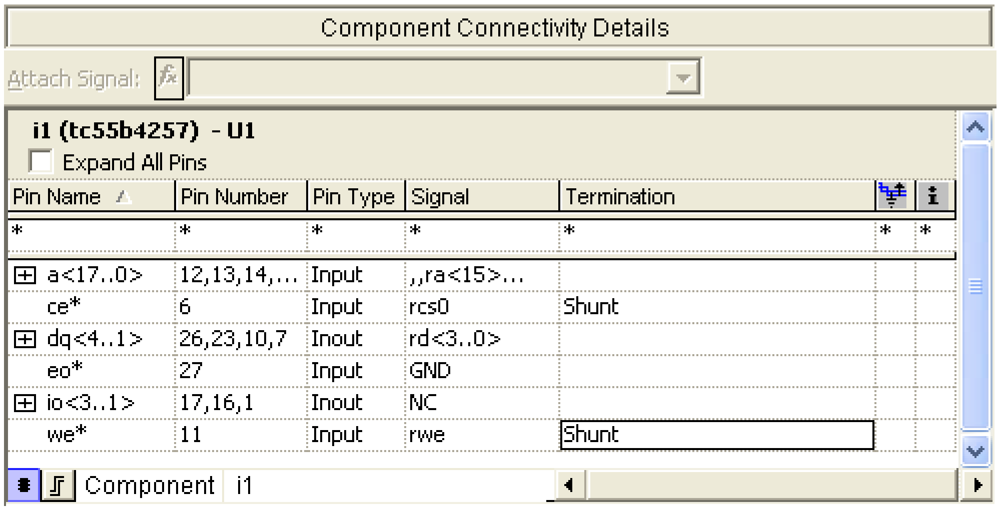
13. Select the shunt termination added on the pin `ce*`.

14. Choose *Edit – Copy* or press *Ctrl + C*.

15. Click on the *Termination* column next to the pin `we*`.

16. Choose *Edit – Paste* or press *Ctrl + V*.

The termination is pasted on the pin we*.



The ability to copy and paste terminations lets you quickly apply terminations on pins in the design.

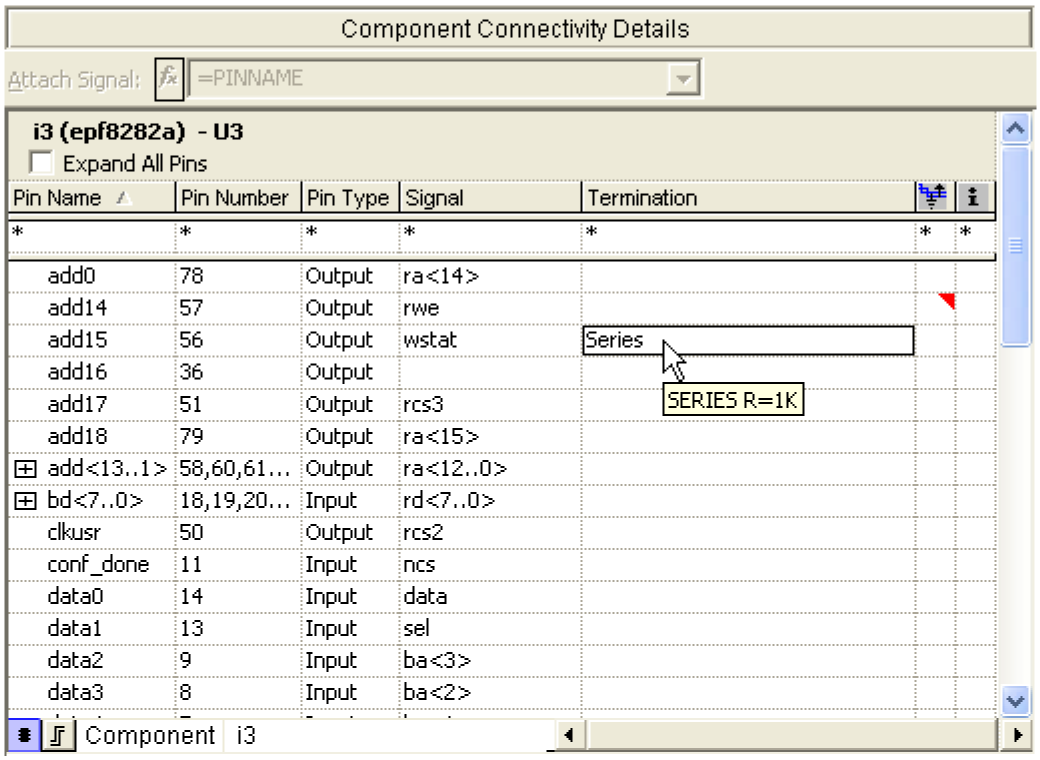
17. Choose *File – Save* to save the design.

Summary

You now know how to add a termination on a pin. You also learned how to copy and paste terminations.

Exercise

Add a series termination on the pin `add15` of the `epf8282a` component as shown below. For the termination use the resistor component `resd` with the value `1K` from the `discrete` library.



For More Information

See:

Working with Associated Components chapter of *Allegro Design Editor User Guide*.

Lesson 3-2: Adding Bypass Capacitors

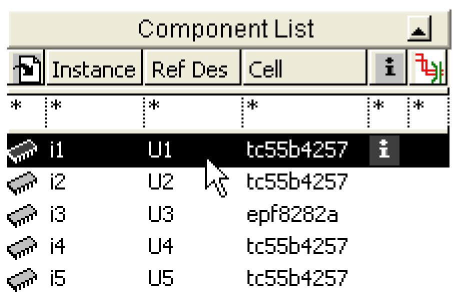
Overview

Bypass capacitors or decoupling capacitors are needed for controlling power and ground bounce in the design.

In this lesson, you will add four bypass capacitors on the memory component `tc55b4257`.

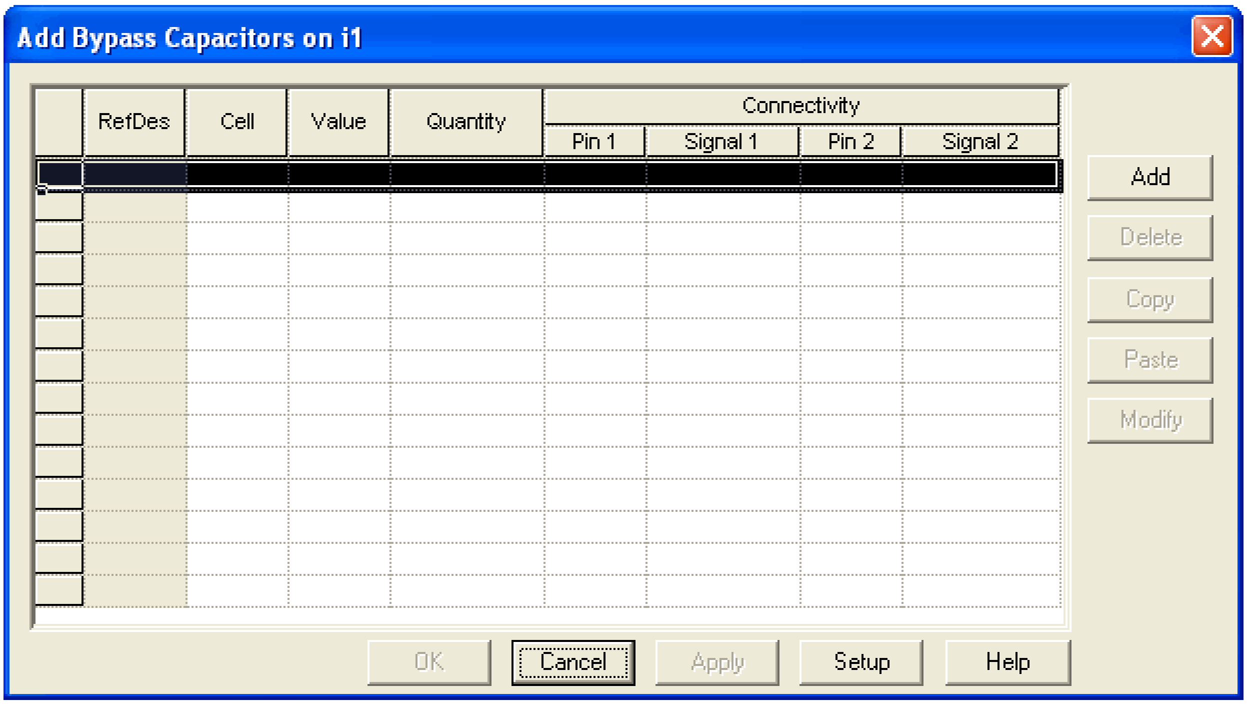
Procedure

1. In the Component List, select instance `i1` of the `tc55b4257` component.



2. Click the right-mouse button and choose *Add Bypass Capacitors* from the shortcut menu.

The Add Bypass Capacitors dialog box appears.

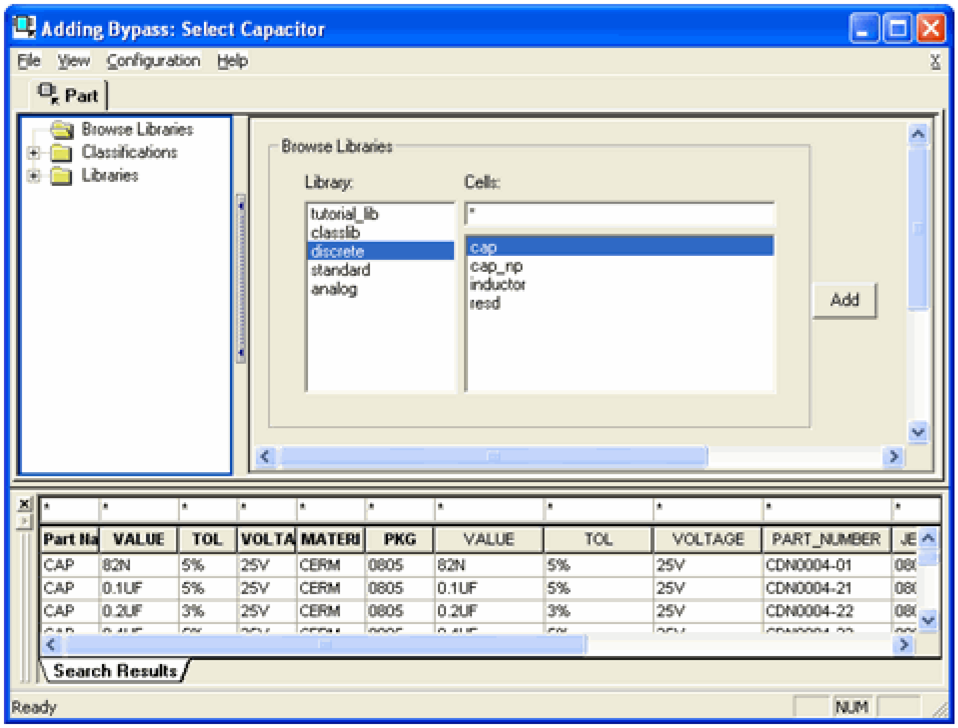


3. Click the *Add* button.

   The Component Browser appears.

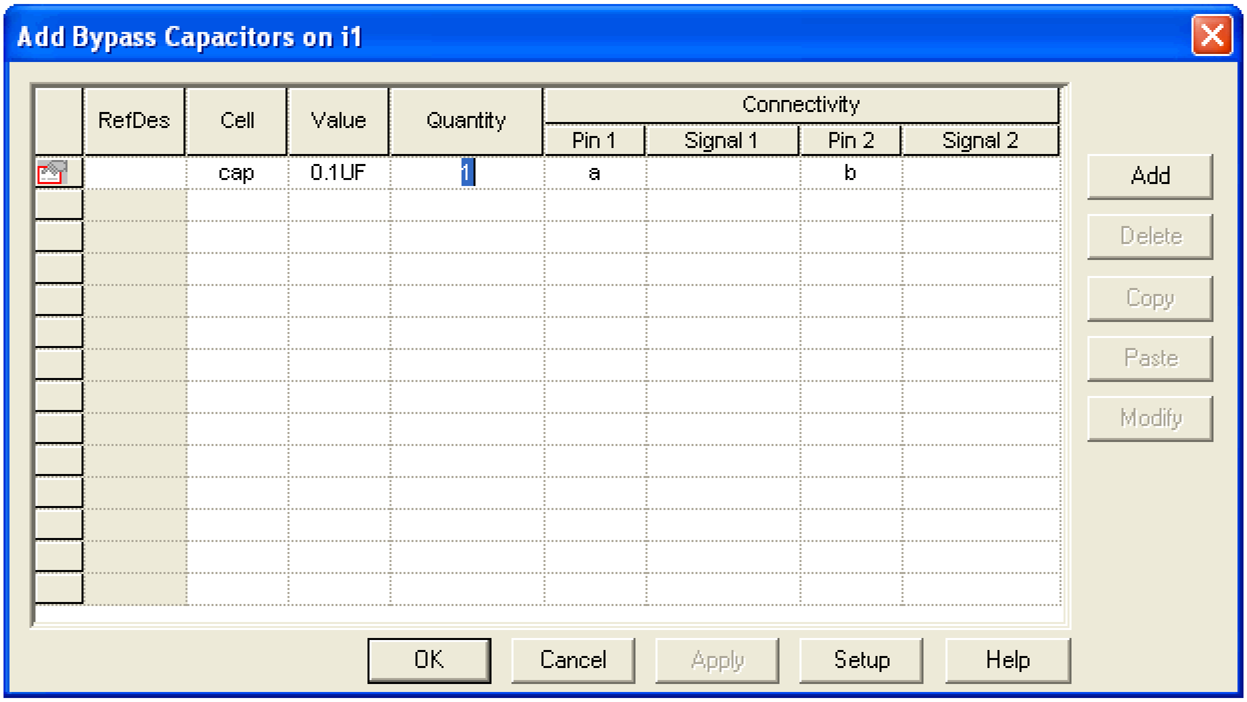
4. In the *Library* list, select the `discrete` library.

5. In the *Cells* list, select the `cap` component.



6. In the *Search Results* pane, select the physical part with the value `0.1UF`.
7. Click *Add*.

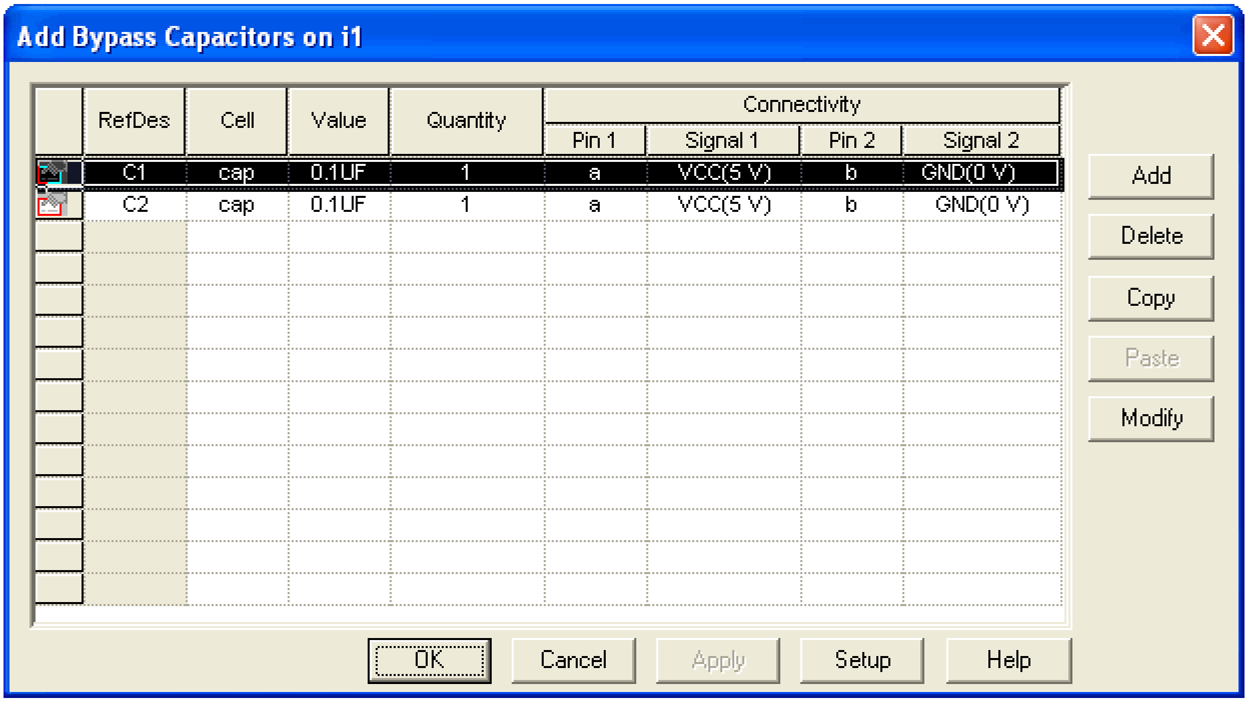
The capacitor component is displayed in the Add Bypass Capacitors dialog box.



You will now add two bypass capacitors between the DC nets `VCC` and `GND`.

8. Change the quantity in the *Quantity* column to 2 and press *Enter*.
9. From the *Signal 1* drop-down list choose `VCC`.
10. From the *Signal 2* drop-down choose `GND`.
11. Click *Apply*.

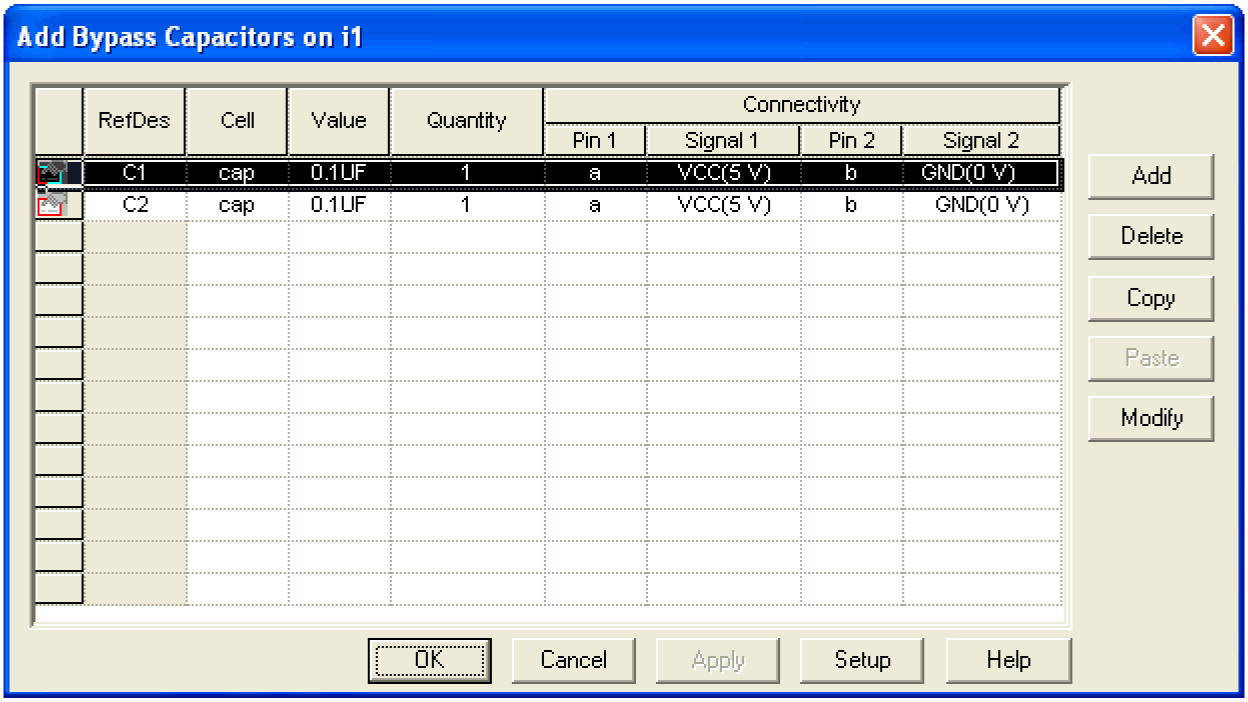
Two bypass capacitors are added on the memory component.



You will now add two more instances of the capacitor component `cap` with the capacitance value of `47UF` as bypass capacitors between the same DC nets.

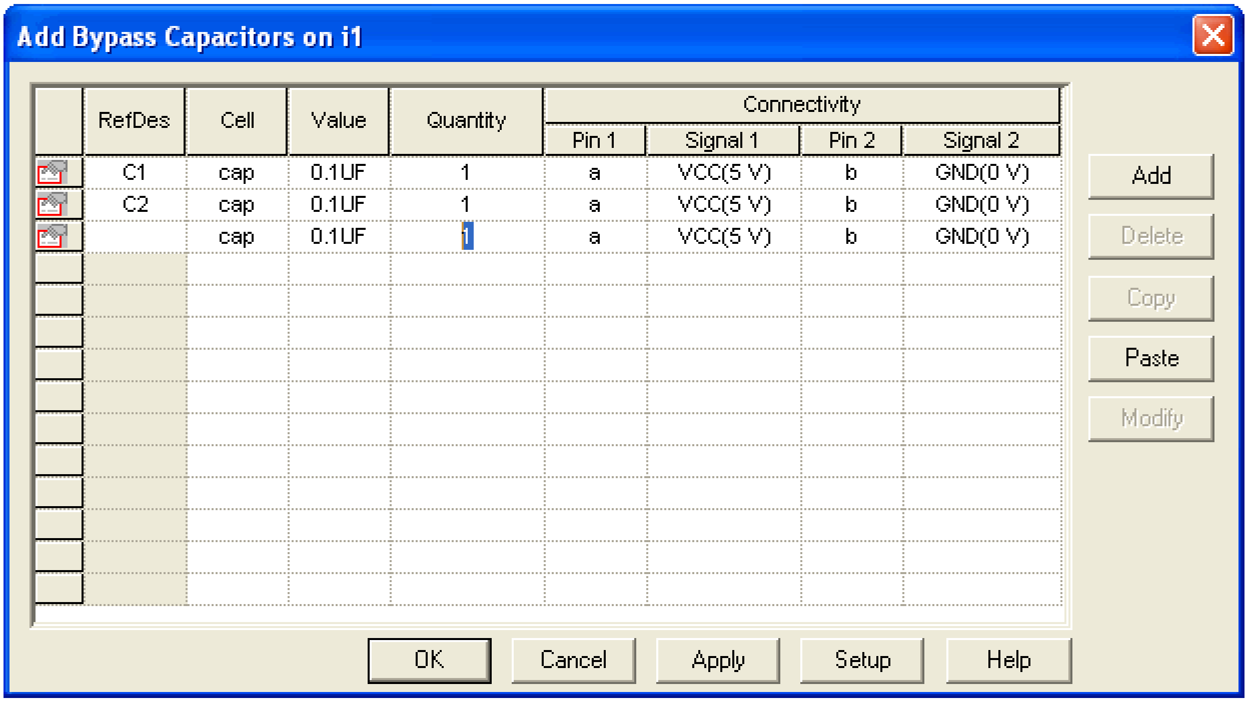
If you want to add more instances of the same capacitor between the same DC nets, but with different capacitance values, you can use the *Copy* and *Paste* buttons in the Add Bypass Capacitors dialog box to copy and paste capacitors, and then use the *Modify* button to modify the capacitance value on the pasted capacitors. This lets you quickly add bypass capacitors on components.

12. Select the first capacitor displayed in the Add Bypass Capacitors dialog box.



13. Click the *Copy* button.

14. Select an empty row in the Add Bypass Capacitors dialog box and click the *Paste* button.

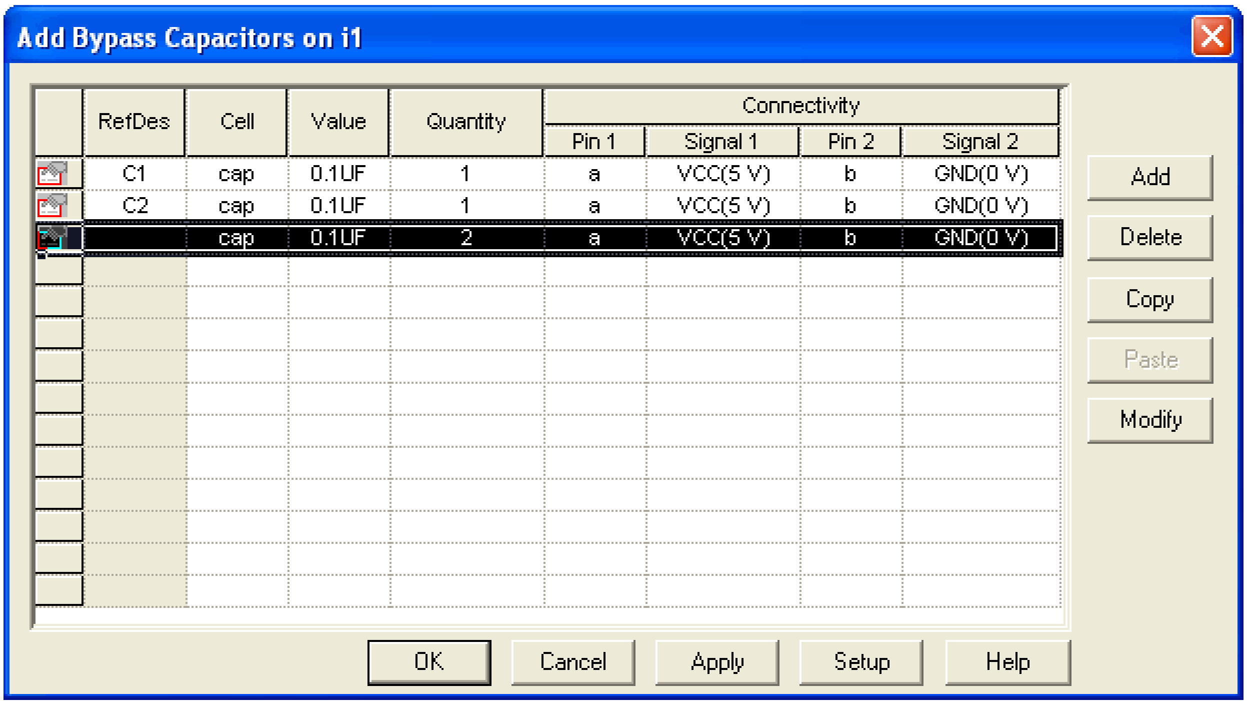


The capacitor is pasted in the Add Bypass Capacitors dialog box.

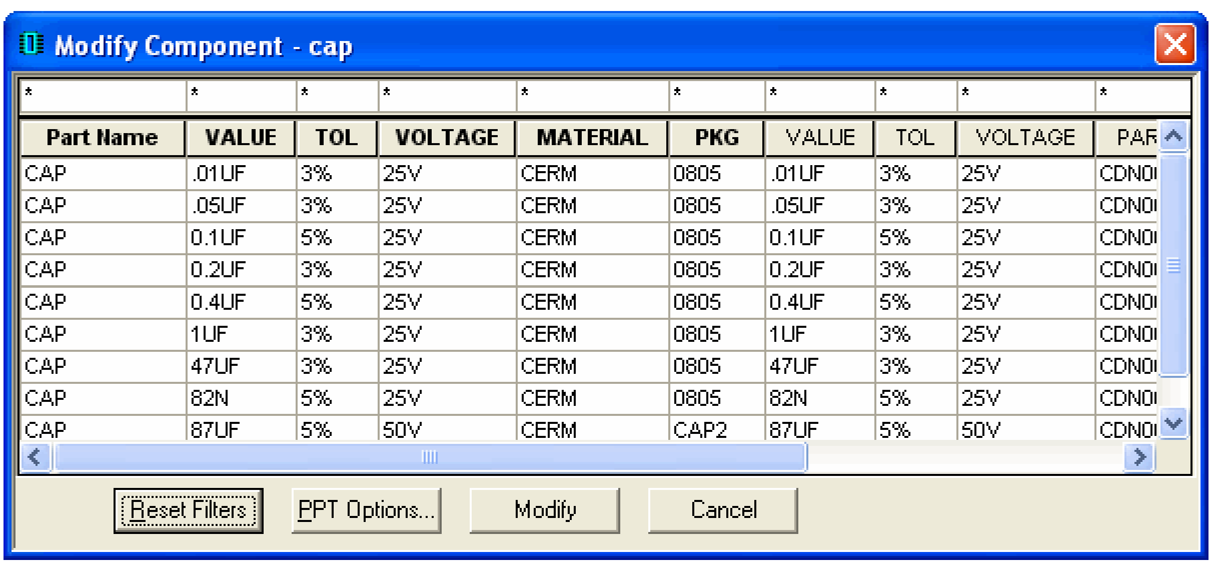
You will now add two capacitors with the value 47UF.

15. Change the quantity in the *Quantity* column to 2 and press *Enter*.

16. Click on the third row and click the *Modify* button.



The Modify Component dialog box appears.

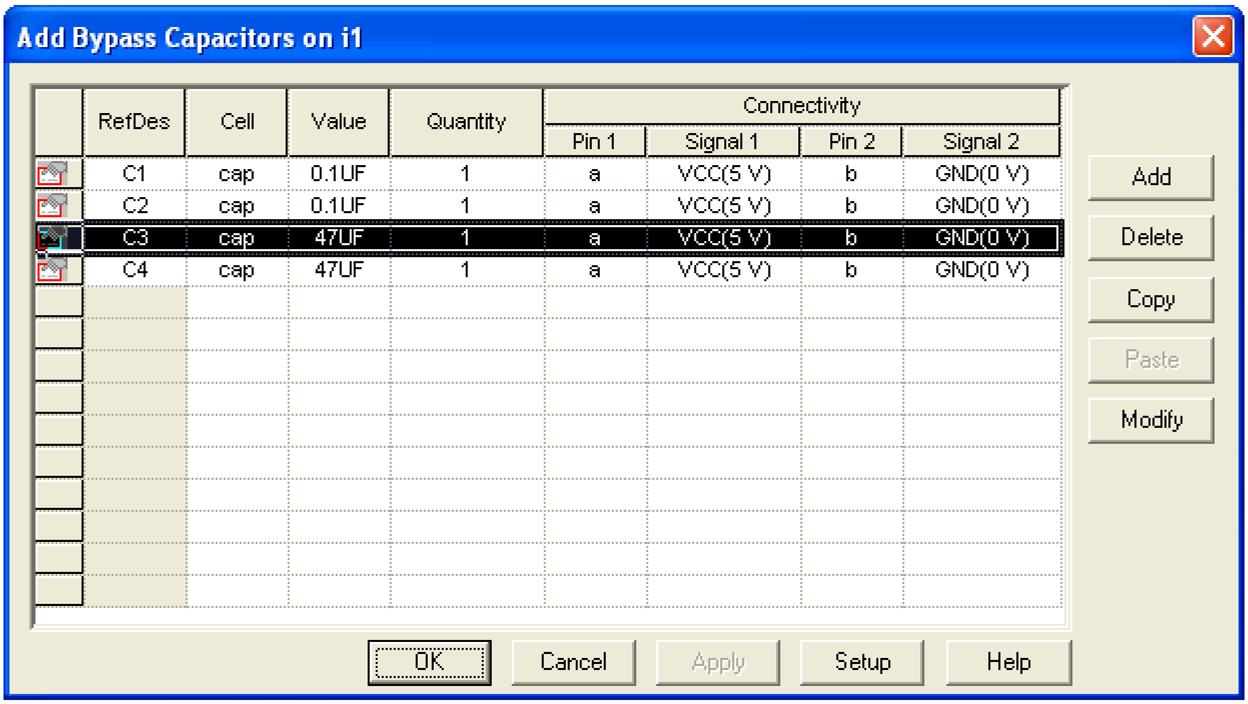


17. Select the part table row with the value 47UF and click the *Modify* button.

The value of the capacitor in the third row is changed to 47UF in the Add Bypass Capacitors dialog box.

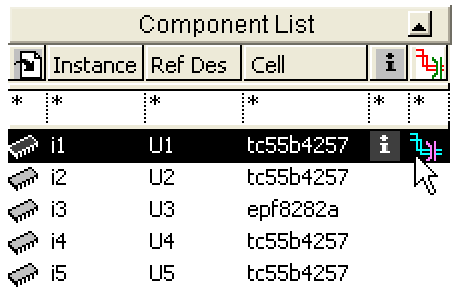
18. Click the *Apply* button.

Two bypass capacitors with the value 47UF are added on the component.



19. Click *OK* to close the Add Bypass Capacitors dialog box.

The  icon next to a component in the Component List indicates that bypass capacitors have been added on the component.



20. Choose *File – Save* to save the design.

Summary

You now know how to add bypass capacitors on a component. You also learned to:

- Copy and paste bypass capacitors
- Modify bypass capacitors
- Identify components in the design on which bypass capacitors are added

For More Information

See:

<u>Working with Associated Components</u> chapter of *Allegro Design Editor User Guide*.

Lesson 3-3: Adding Pullups and Pulldowns

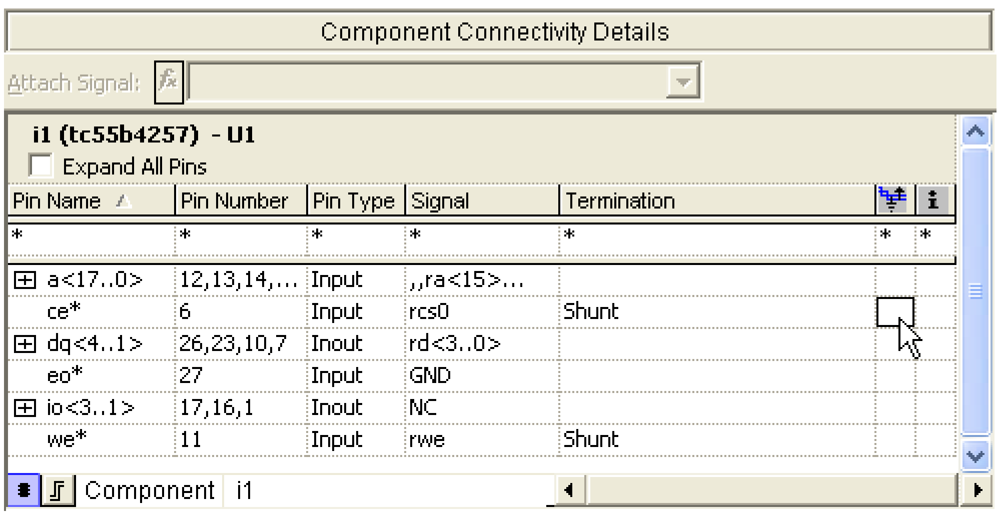
Overview

It is recommended to pullup or pulldown open pins of a component to reduce noise in the circuit.

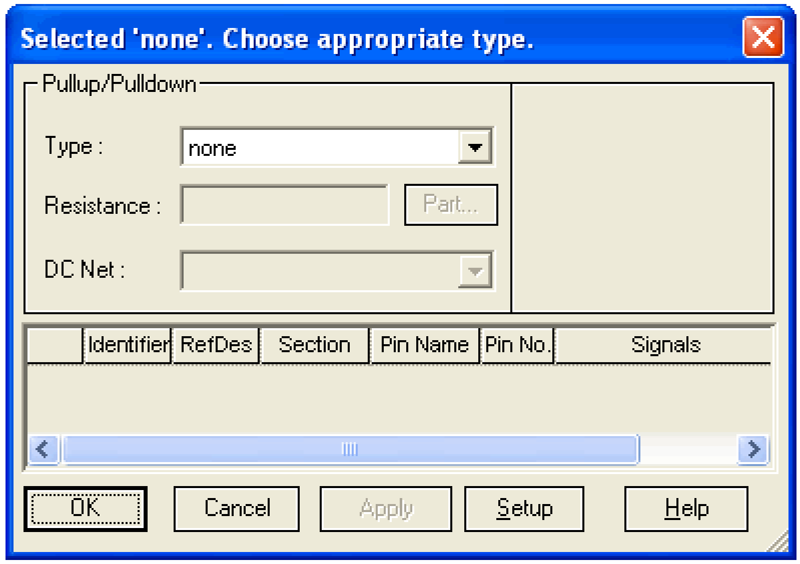
In this lesson, you will learn to add a pullup on a net.

Procedure

1. In the Component List, select instance `i1` of the `tc55b4257` component.
2. In the Component Connectivity Details pane, double-click in the Pullup/Pulldown column (the column with the icon) next to the `ce*` pin.

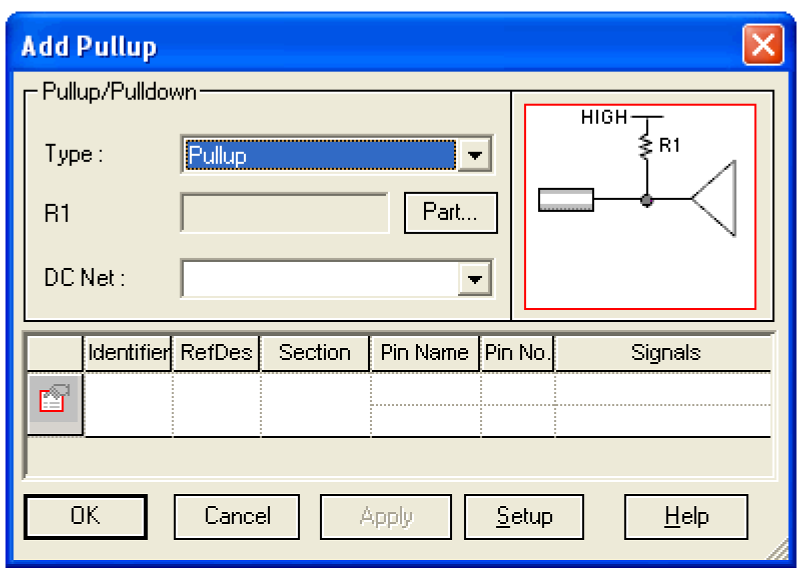


The Add Pullup/Pulldown dialog box appears.



3. From the Type drop-down list, choose *Pullup* to pullup the pin ce*.

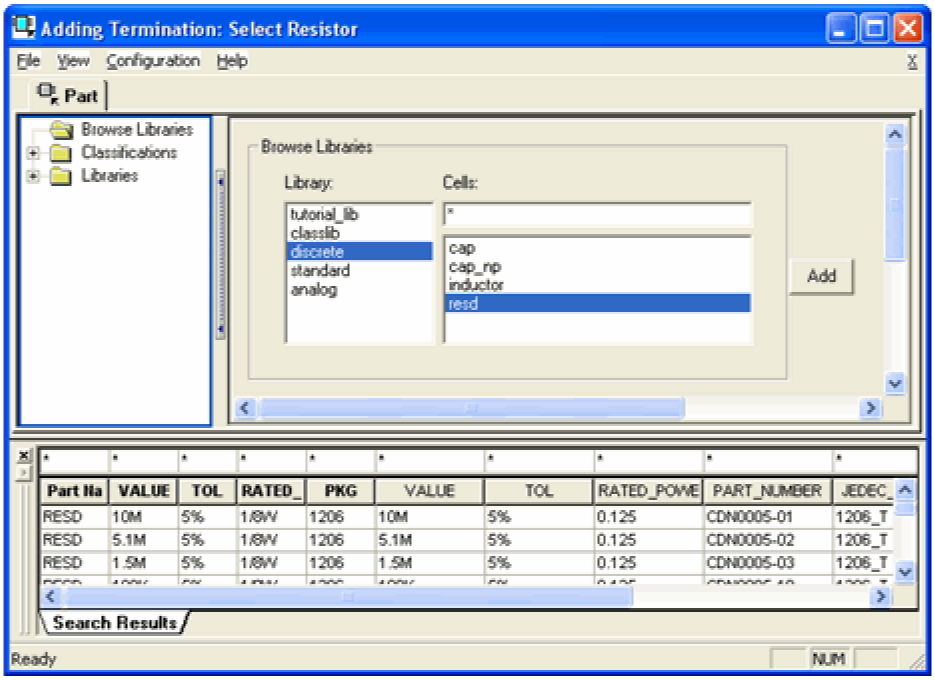
   The Add Pullup dialog box displays a graphical representation of the pullup scheme.



4. Click the *Part* button to select the resistor you want to use for the pullup.

The Component Browser appears.

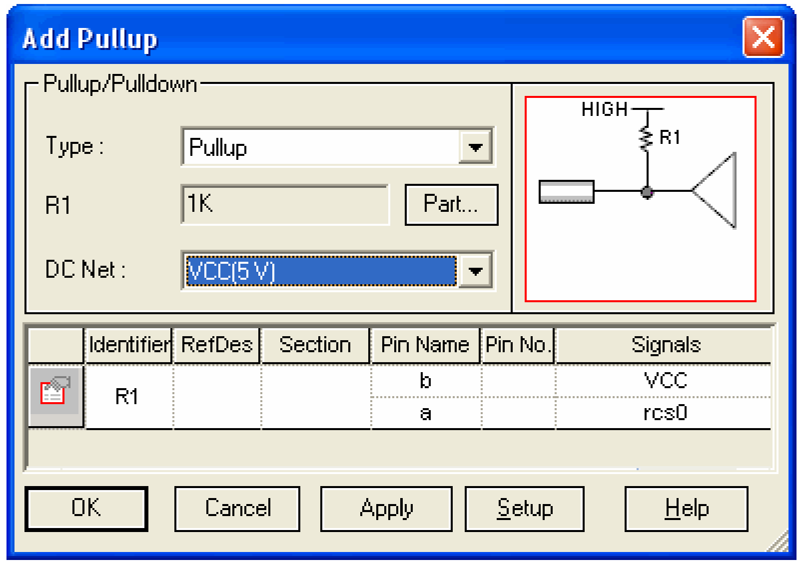
5. In the *Library* list, select the `discrete` library.
6. In the *Cells* list, select the `resd` component.



7. In the *Search Results* pane, select the physical part with the value `1K`.
8. Click the *Add* button.

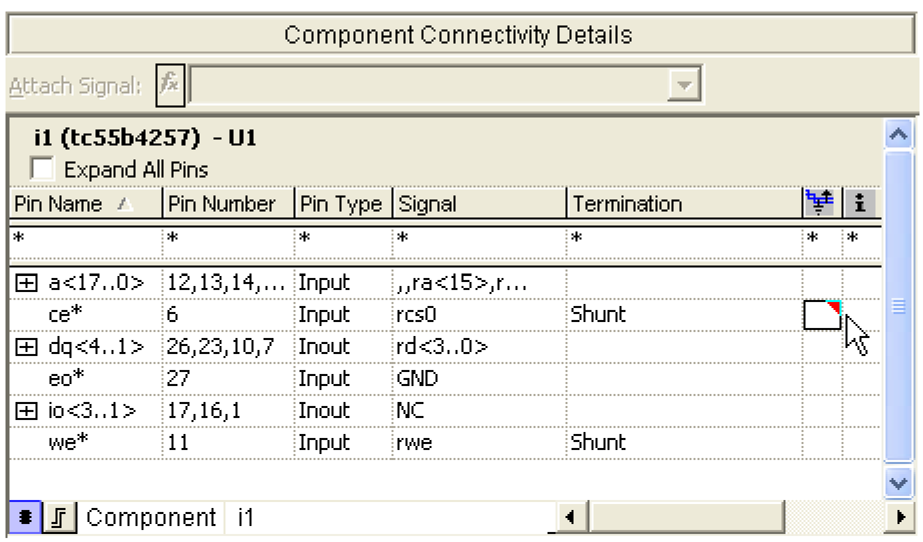
   The value of the resistor is displayed in the *R1* field in the Add Pullup dialog box.

9. From the *DC Net* drop-down list, select the net VCC.



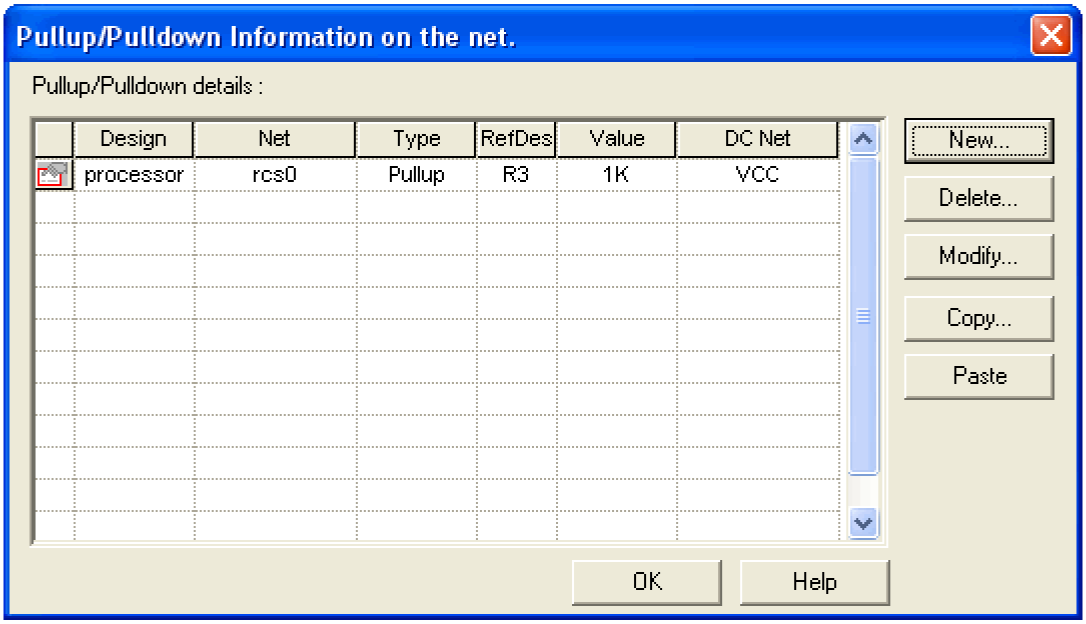
10. Click *OK* to add the pullup.

    The red triangle at top-right of the cell in the Pullup/Pulldown column (the column with the icon) indicates that a pullup is attached to the net.



11. Double-click on the cell containing the pullup in the Pullup/ Pulldown column.

The Pullup/Pulldown information on the net dialog box appears displaying the details of the pullup on the net.



Here you can add, copy/paste, modify and delete pull-ups and pulldowns on the net.

12. Click *OK* to close the Pullup/Pulldown information on the net dialog box.

13. You can copy pullups and pulldowns from one net and paste it on another net.

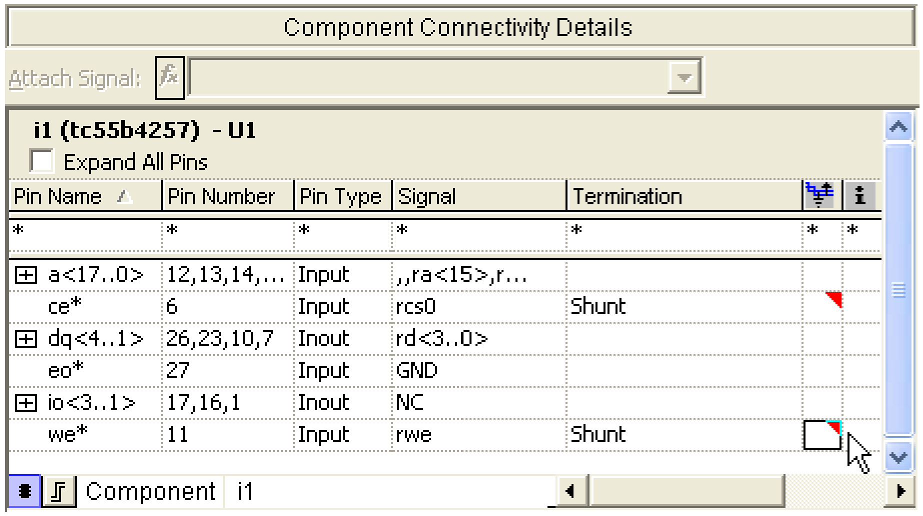
   a. Click on the cell containing the pullup in the Pullup/Pulldown column (the column with the icon).

   b. Choose *Edit – Copy* or press *Ctrl + C.*

   c. Click on Pullup/Pulldown column next to the `rwe` net.

   d. Choose *Edit – Paste* or press *Ctrl + V.*

The pullup is pasted on the `rwe` net.



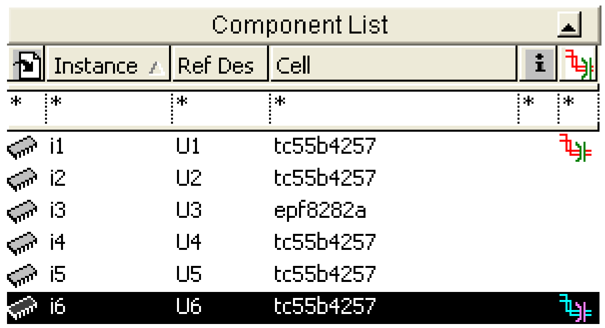
14. When you copy and paste a component in the Component List, its connectivity (pin-signal connectivity and the terminations, bypass capacitors and pullup/pulldowns), and property information are also pasted on the new instance of the component. This feature lets you capture connectivity and property information on one instance of a component and quickly replicate the connectivity and property information on additional instances of the same component you want to use in your design.

a. In the Component List, select instance `i1` of the `tc55b4257` component.

b. Choose *Edit – Copy* or press *Ctrl + C.*

c. Choose *Edit – Paste* or press *Ctrl + V.*

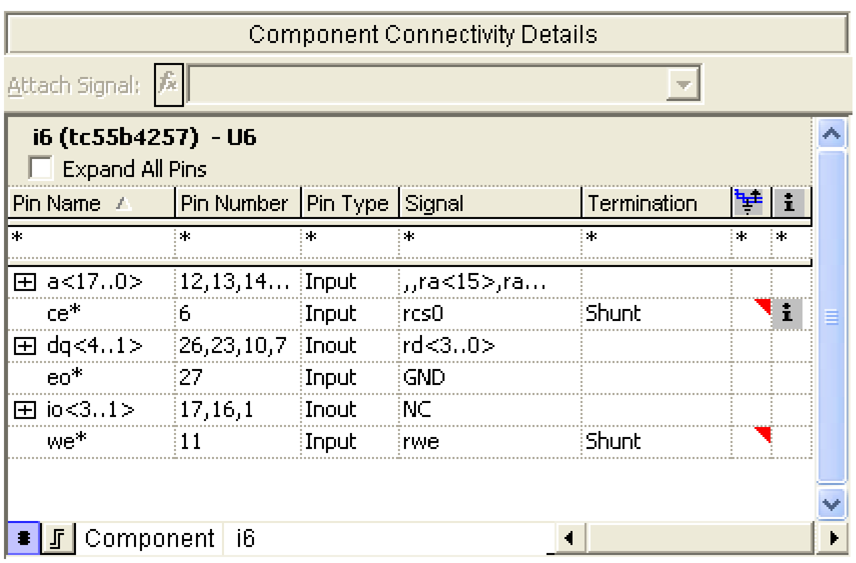
A new instance of the `tc55b4257` component with the instance name `i6` is added in the Component List.



Note that the bypass capacitors on the original component has been copied and pasted on the new instance of the component.

d. Select instance `i6` of the `tc55b4257` component in the Component List.

The connectivity for the component is displayed in the Component Connectivity Details pane.



Note that the connectivity information (pin-signal connectivity and the terminations, bypass capacitors and pullup/pulldowns) on the original component has also been pasted on the new instance of the component.

15. Choose *File – Save* to save the design.

Summary

You now know how to add pullups on nets and how to copy a pullup added on a net and paste it on another net.

You also learned that when you copy and paste a component in the Component List, its connectivity information (pin-signal connectivity, terminations, bypass capacitors and pullup/pulldowns) is also pasted on the new instance of the component.

For More Information

See:

Working with Associated Components chapter of *Allegro Design Editor User Guide*.

Lesson 3-4: Using the Associated Component Viewer

Overview

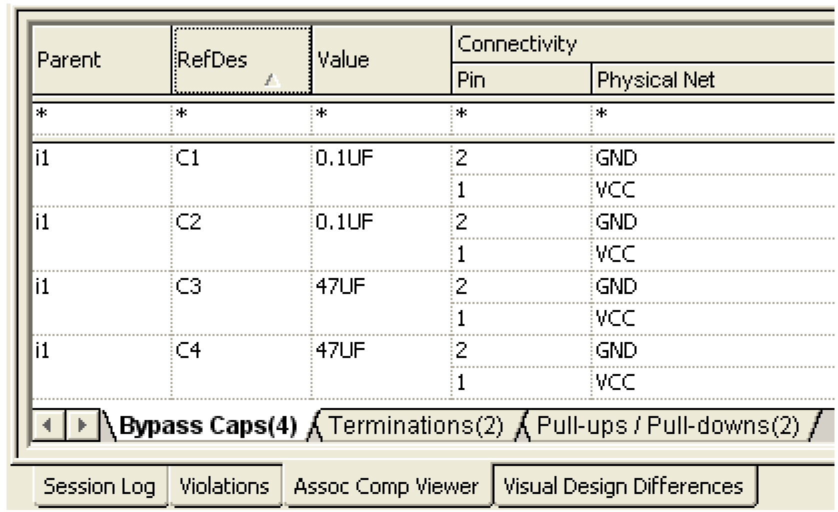
The individual discrete components attached to an object and the nets created as a result of adding terminations, bypass capacitors and pullup/pulldowns are displayed in the Associated Component Viewer and not in the Component List and the Signal List. This ensures that the design is not cluttered and keeps you focused on capturing the design's logic.

In this lesson you will learn to view the associated components added on a component.

Procedure

1. Select instance `i1` of the `tc55b4257` component in the Component List.
2. Choose *View – Associated Components*.

The *Assoc Comp Viewer* tab displays the details of the bypass capacitors on the component.



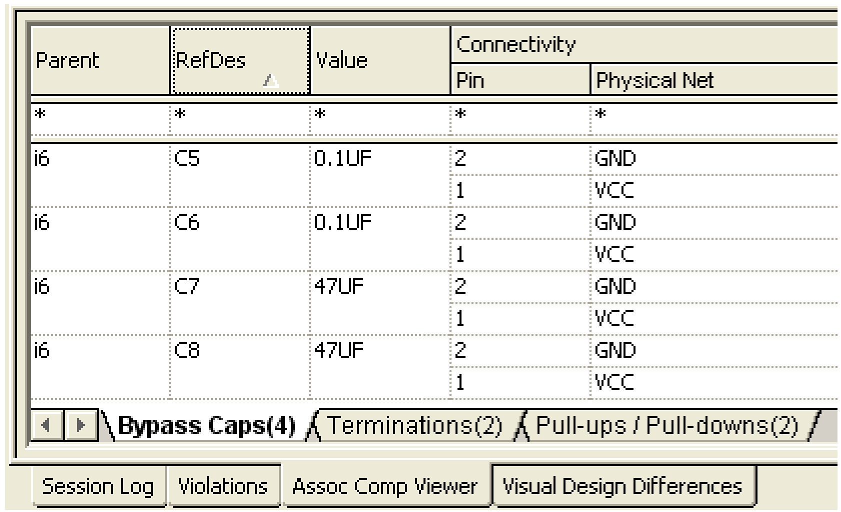
The Bypass Caps, Terminations and Pull-ups/Pull-downs tabs in the Assoc Comp Viewer display the number of terminations, bypass capacitors and pullup/pulldowns on the component.

3. Click on the Terminations and Pull-ups/Pull-downs tabs to view the details of the terminations and pullups and pulldowns added on the component.

4. In the Associated Component Viewer, you can modify the value of discrete components (resistors, capacitors, inductors and diodes) used in terminations, bypass capacitors and pullups and pulldowns

a. Select instance `i6` of the `tc55b4257` component in the Component List.

The *Assoc Comp Viewer* tab displays the details of the bypass capacitors on the component.

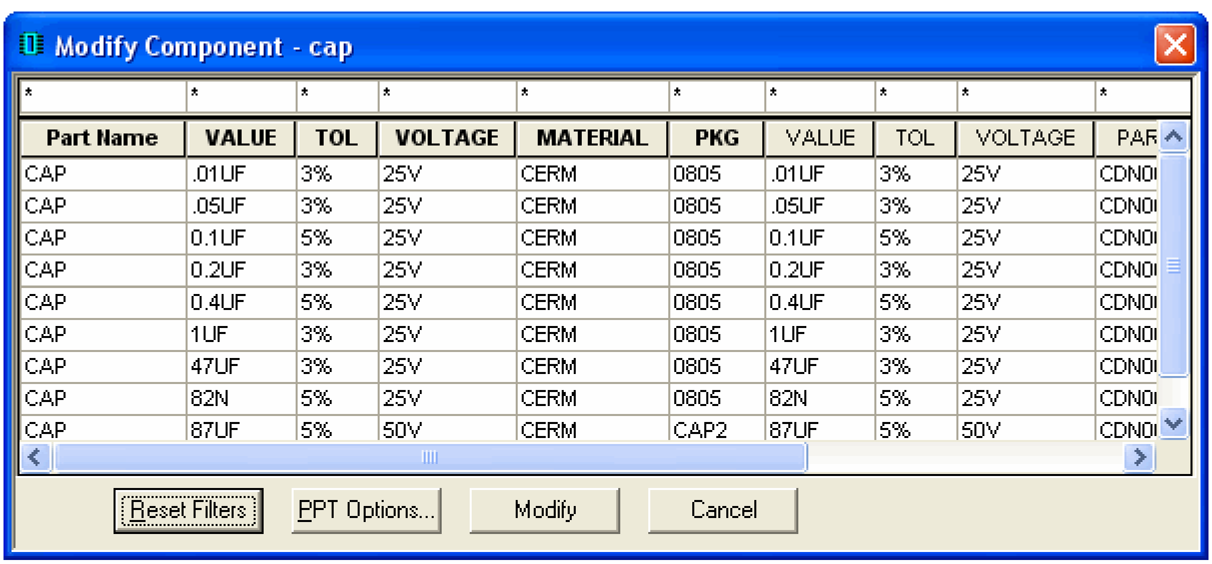


Parent	RefDes	Value	Connectivity	
			Pin	Physical Net
*	*	*	*	*
i6	C5	0.1UF	2	GND
			1	VCC
i6	C6	0.1UF	2	GND
			1	VCC
i6	C7	47UF	2	GND
			1	VCC
i6	C8	47UF	2	GND
			1	VCC

b. Click on the capacitor with the reference designator `C7`.

c. Click the right-mouse button and choose *Modify Component*.

The Modify Component dialog box appears.



*	*	*	*	*	*	*	*	*	*
Part Name	**VALUE**	**TOL**	**VOLTAGE**	**MATERIAL**	**PKG**	VALUE	TOL	VOLTAGE	PAR
CAP	.01UF	3%	25V	CERM	0805	.01UF	3%	25V	CDN0
CAP	.05UF	3%	25V	CERM	0805	.05UF	3%	25V	CDN0
CAP	0.1UF	5%	25V	CERM	0805	0.1UF	5%	25V	CDN0
CAP	0.2UF	3%	25V	CERM	0805	0.2UF	3%	25V	CDN0
CAP	0.4UF	5%	25V	CERM	0805	0.4UF	5%	25V	CDN0
CAP	1UF	3%	25V	CERM	0805	1UF	3%	25V	CDN0
CAP	47UF	3%	25V	CERM	0805	47UF	3%	25V	CDN0
CAP	82N	5%	25V	CERM	0805	82N	5%	25V	CDN0
CAP	87UF	5%	50V	CERM	CAP2	87UF	5%	50V	CDN0

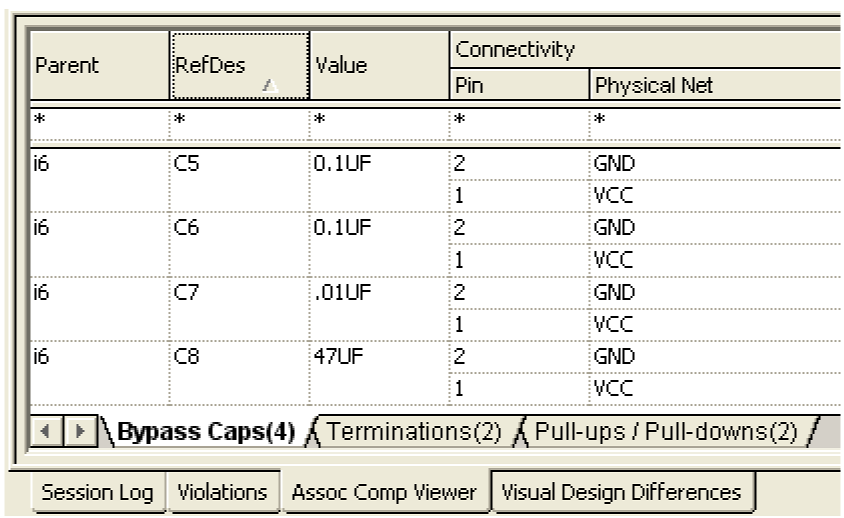
5. Select the part table row with the value `.01uf` and click the *Modify* button.

The value of the capacitor is changed to `.01uf` in the *Bypass Caps* tab of the Assoc Comp Viewer.

6. The discrete components (resistors, capacitors, diodes and inductors used in terminations, bypass capacitors and pullup/pulldowns) are directly associated with the objects on which you add the terminations, bypass capacitors or pullup/pulldowns. If you delete the object, the discrete components associated with the object are also automatically deleted. This makes it very easy for you to manage discrete components in your design.

a. Select instance `i6` of the `tc55b4257` component in the Component List.

The *Assoc Comp Viewer* tab displays the details of the bypass capacitors on the component.



Parent	RefDes	Value	Connectivity	
			Pin	Physical Net
*	*	*	*	*
i6	C5	0.1UF	2	GND
			1	VCC
i6	C6	0.1UF	2	GND
			1	VCC
i6	C7	.01UF	2	GND
			1	VCC
i6	C8	47UF	2	GND
			1	VCC

b. Press *Delete*.

The component and its associated discrete components are deleted from the design.

7. Choose *File – Save* to save the design.

Summary

You now know how to view the associated components added on components using the Associated Component Viewer. You also learned the following:

- How to modify the value of associated components
- Deleting a component in the Component List deletes the associated components connected to it. This makes it easy to manage associated components in the design as you do not have to manually delete the associated components connected to a component when you delete the component in the design.

For More Information

See:

Working with Associated Components chapter of *Allegro Design Editor User Guide*.

Module 4: Working with Properties

Prerequisite

If you have not completed all the lessons in the previous modules, you must open the `tutorial.cpm` project located at `<your_work_area>\modules\properties\tutorial` in Design Editor and perform the steps described in this module.

For more information, see Understanding the Sample Design Files on page 14.

Lessons

This module consists of the following lessons:

Completion Time

1 hour for written lessons

Overview

Properties (also called attributes) are used to convey information about a design. Properties carry such information as the part number of a component, the voltage of a net and so on. Properties consist of a name and value. Some properties have a set of standard values that you can use; other properties support any value that you assign.

Properties added on components and nets in the design can also affect component placement and signal routing. This lets you specify physical layout requirements in the design capture stage itself.

Design Editor supports a predefined set of properties that you can add on design objects (components, nets or pins). You can also define custom (user-defined) properties and add them on design objects.

You can use the Properties window in Design Editor to work with properties on individual components, nets and pins in your design. You can also use Constraint Manager to capture and manage property information in your design. Constraint Manager provides a spreadsheet interface that helps you to quickly work with properties across your design.

In this module you will learn to work with properties in Design Editor and in Constraint Manager.

Lesson 4-1: Working with Properties in the Properties Window in Design Editor

Overview

The Properties window in Design Editor allows you to work with properties on individual components, nets and pins in your design.

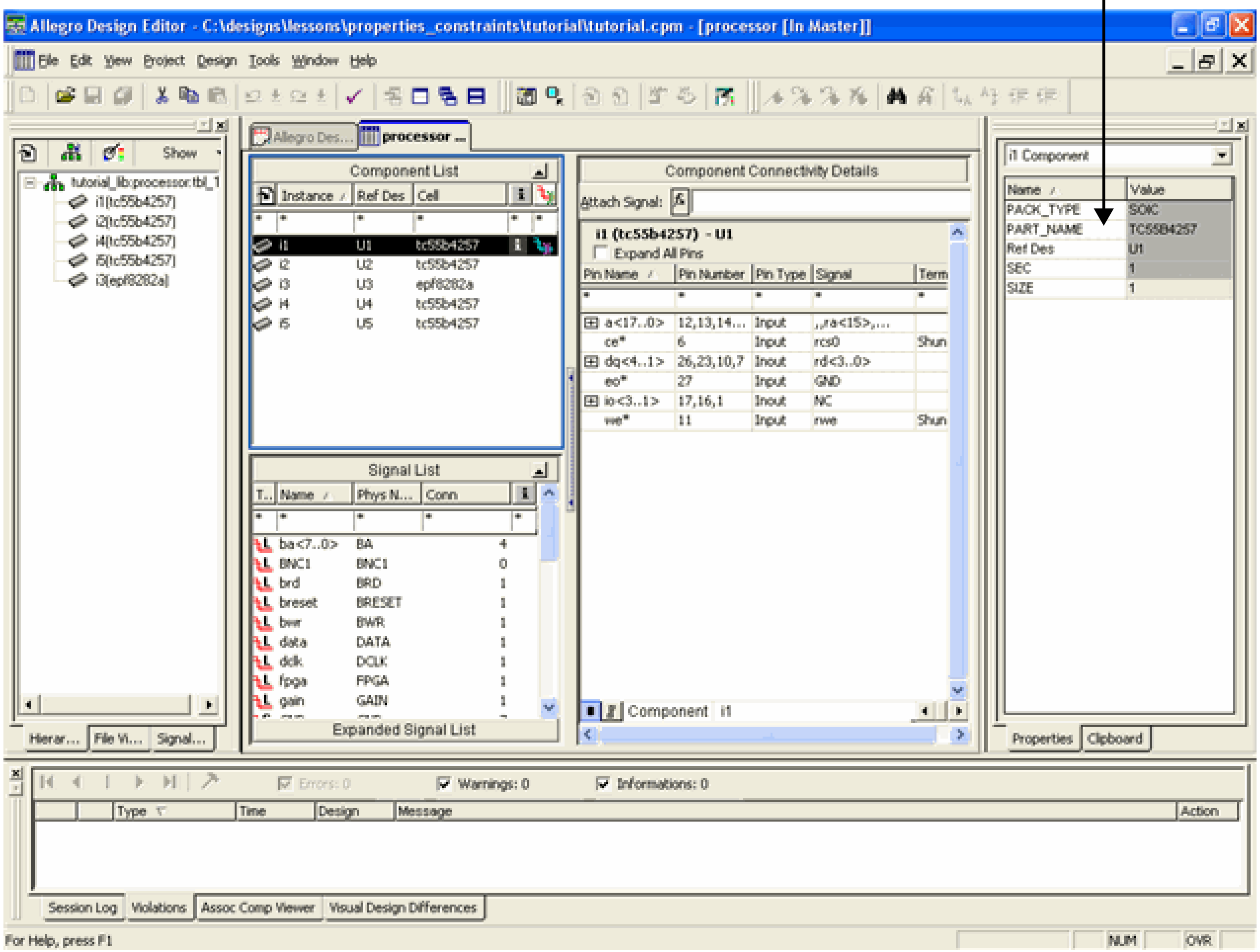
In this lesson, you will learn to work with properties in the Properties window in Design Editor.

Procedure

1. Choose *View – Properties Window*.

The Properties window appears on the right side of the Design Editor workspace.

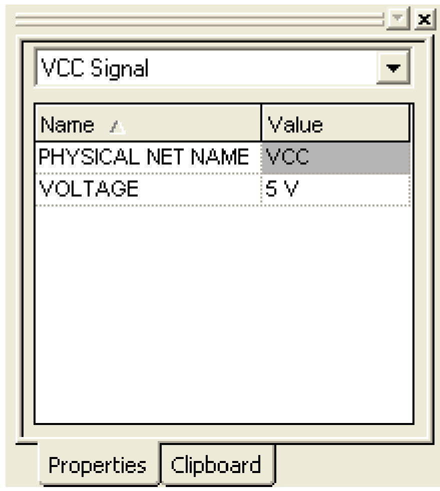
Properties Window



The Properties window displays the properties on instance `i1` of the `tc55b4257` component because the instance is selected in the Component List.

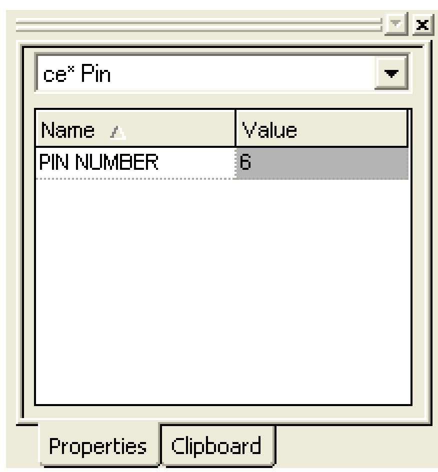
2. Select the VCC signal in the Signal List.

The properties on the VCC signal are displayed in the Properties window.



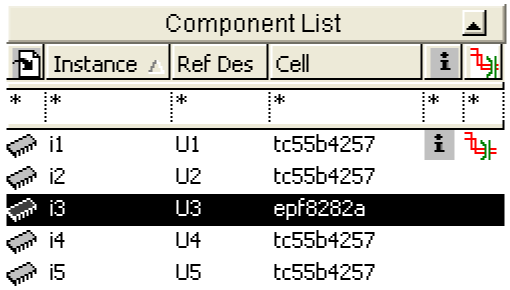
3. Select the `ce*` pin in the Component Connectivity Details pane.

   The properties on the `ce*` pin are displayed in the Properties window.

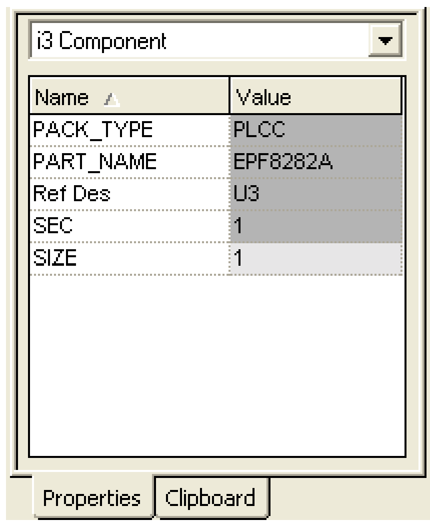


You will now add the `ROOM` property with the value `FPGA` on the `epf8282a` component. The `ROOM` property lets you control where parts are placed in the Allegro PCB Editor board.

4. In the Component List, select the `epf8282a` component.

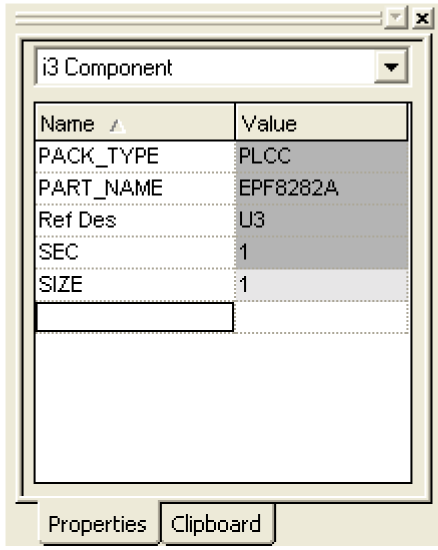


The properties on the component are displayed in the Properties window.



5. Click the right-mouse button in the Properties window and choose *Insert Property* from the shortcut menu.

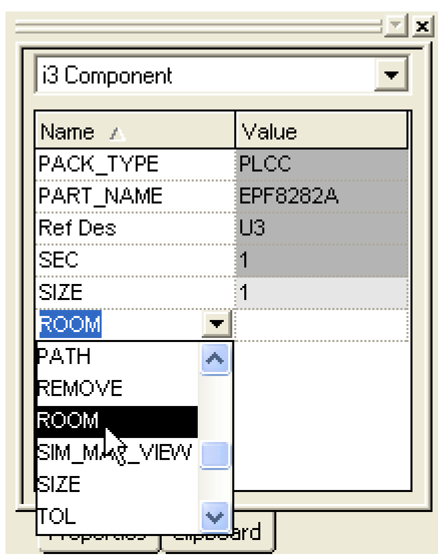
A new row is added in the Properties window.



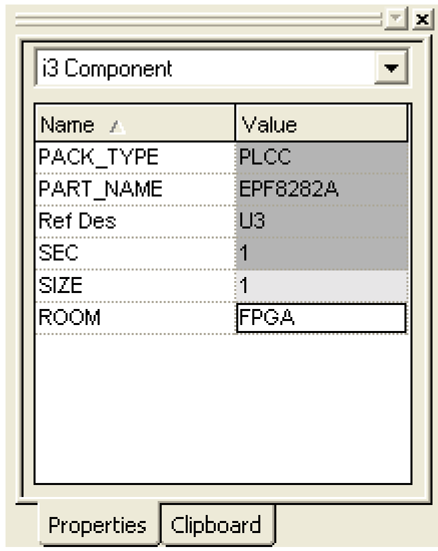
Tip

You can also press the *Insert* key to add a property.

6. Click in the *Name* column in the new row and choose `ROOM` from the drop-down list.



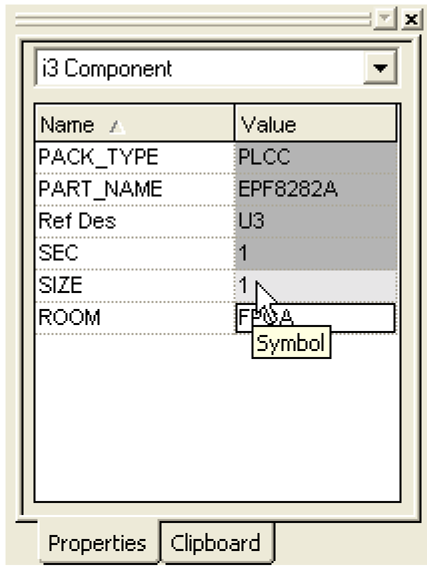
7. In the *Value* column, enter `FPGA` as the value for the `ROOM` property.



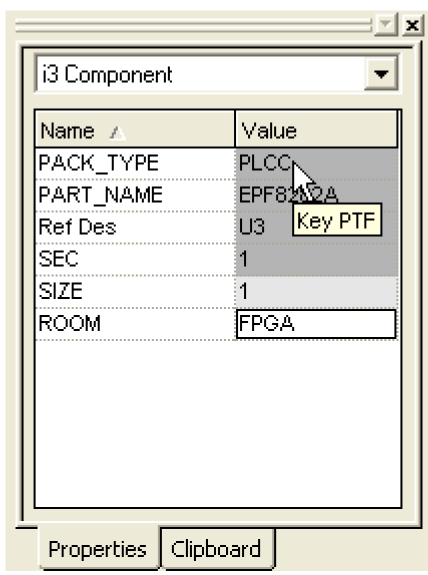
Similarly, you can select a signal in the Signal List or a pin in the Component Connectivity Details pane to display its properties in the Properties window and then add properties on the signal or pin.

Note: You can copy property values from another component, signal or pin, or from another application such as Microsoft Excel, and paste it in the Properties window.

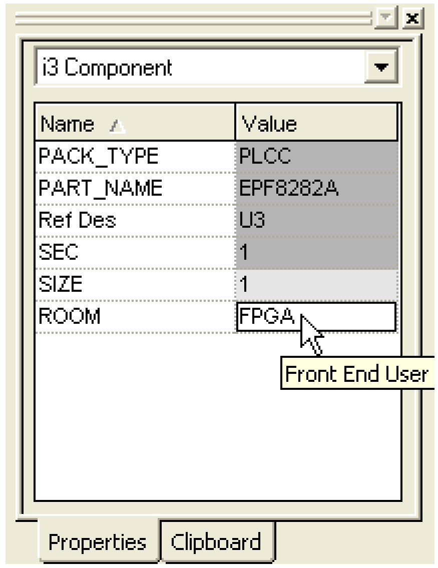
Note that the values for the `PACK_TYPE`, `PART_NAME`, `Ref Des`, `SEC` and `SIZE` properties are grayed out in the Properties window. You cannot modify the values for these properties because they are added on the symbol for the component, annotated from the physical part table, or automatically assigned by Design Editor. If you place the mouse pointer over a property value, a tooltip appears displaying the origin of the property.



In the above figure, the tooltip `Symbol` indicates that the `SIZE` property exists on the symbol for the `epf8282a` component.

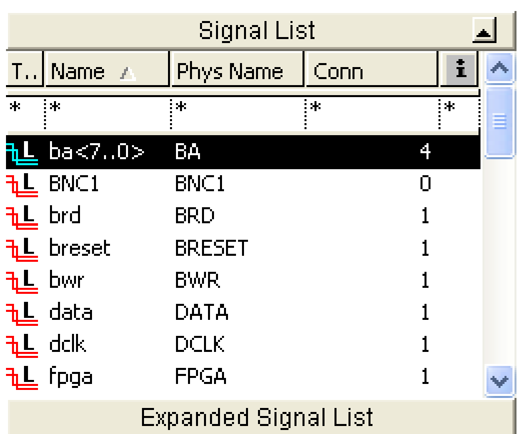


In the above figure, the tooltip `Key PTF` indicates that the PACK_TYPE property is a key property that is annotated from the physical part table file. For more information on physical part table files, see Lesson 1-3: Setting Up the Project on page 41.

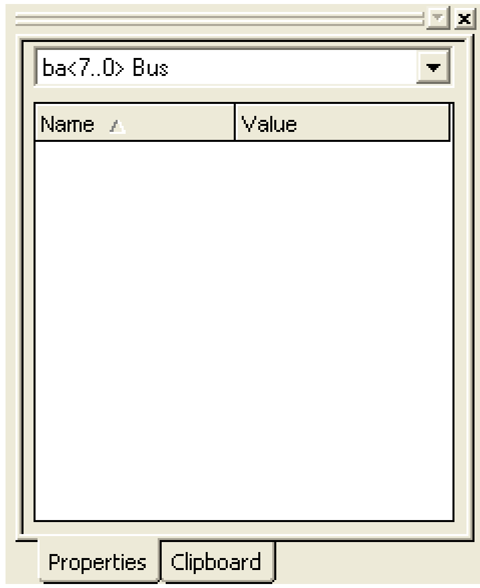


In the above figure, the tooltip `Front End User` indicates that the `ROOM` property is added by you in Design Editor.

8. In the Signal List, select the vectored signal (bus) `ba<7..0>`.

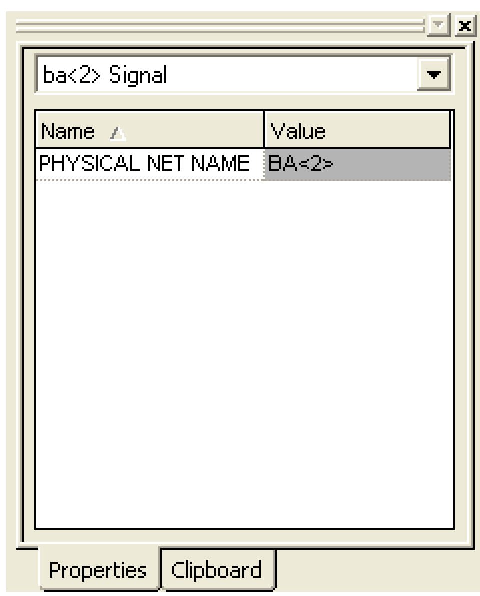


The Properties window appears as shown below.



Note: You cannot add properties on the vectored signal. You can only add properties on the bits of the vectored signal.

9. Click the drop-down list and select the bit ba<2>.



The properties on the bit are displayed in the Properties window. You can add properties on the bit.

Summary

You now know how to use the Properties window in Design Editor to view and add properties on components, signals and pins in the design.

For More Information

See:

Working with Properties and Electrical Constraints chapter of *Allegro Design Editor User Guide*.

Lesson 4-2: Working with Properties in Constraint Manager

Overview

Constraint Manager allows you to capture and manage property information in your design. Constraint Manager provides a spreadsheet interface that helps you to quickly work with properties across your design.

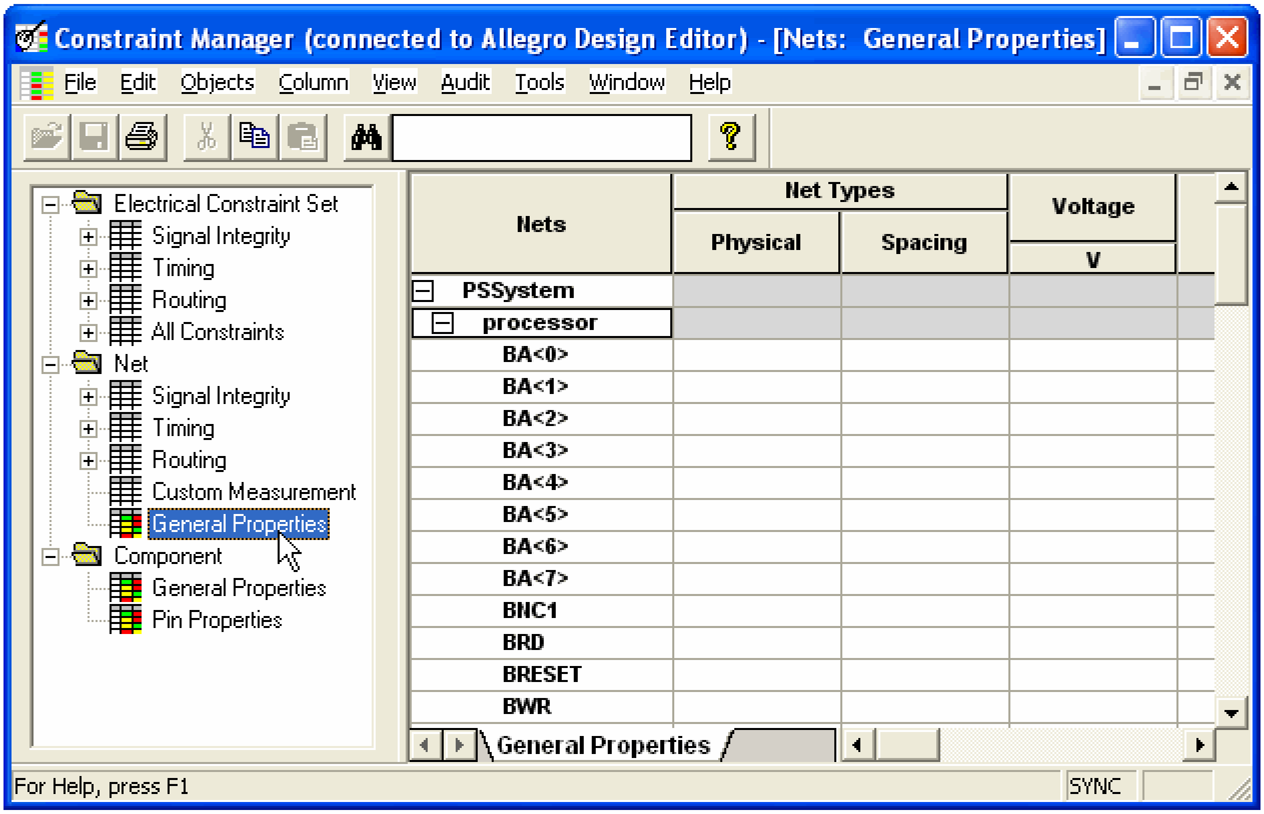
In this lesson, you will learn to work with properties in Constraint Manager.

Procedure

1. To open Constraint Manager from Design Editor, do one of the following:
   - ❑ Click the  toolbar button.
   - ❑ Choose *Design – Edit Constraints*.

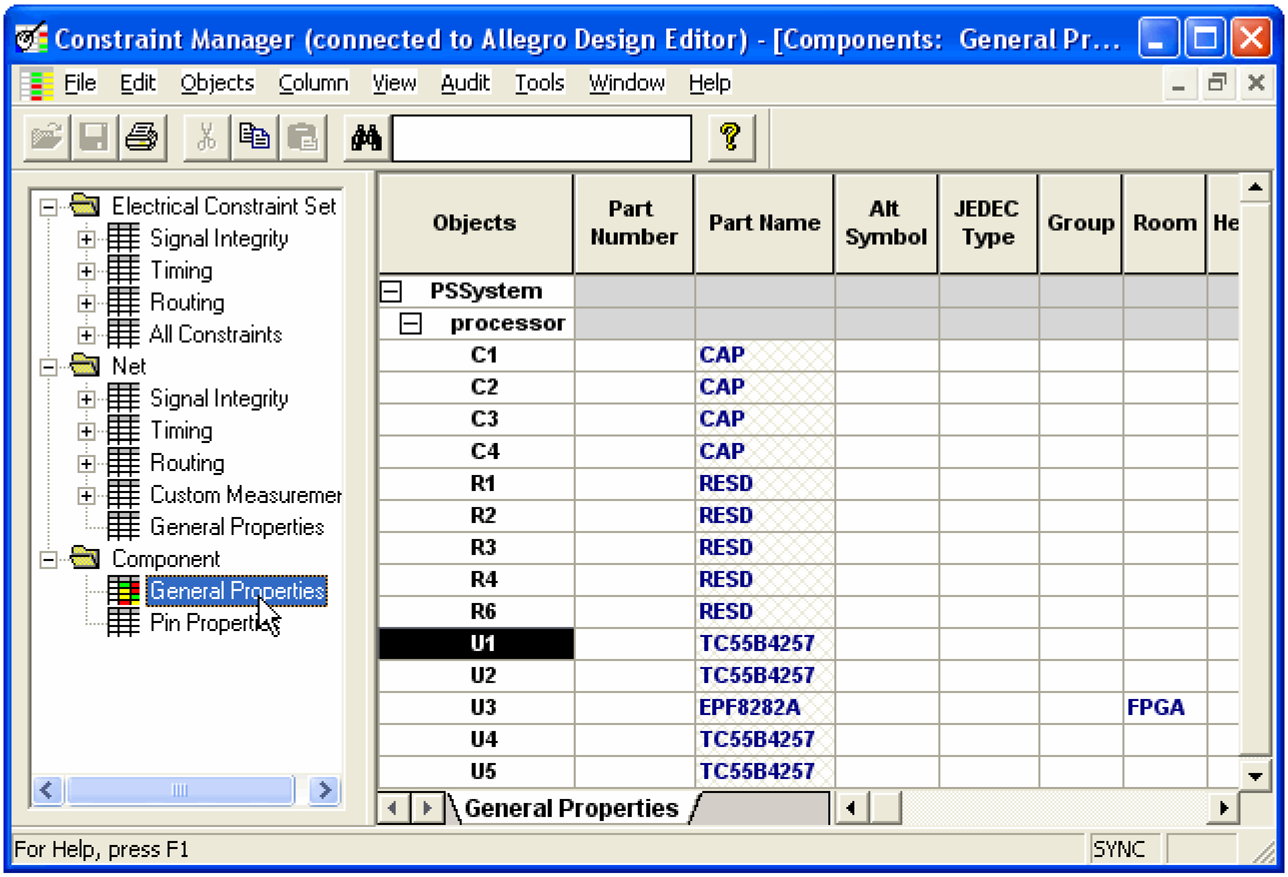
   The Constraint Manager window appears.

The *General Properties* workbook in the *Net* object folder lets you work with properties on nets in the design.



Note that nets are displayed using physical net names in Constraint Manager.

The *General Properties* workbook in the *Component* object folder lets you work with properties on components in the design.

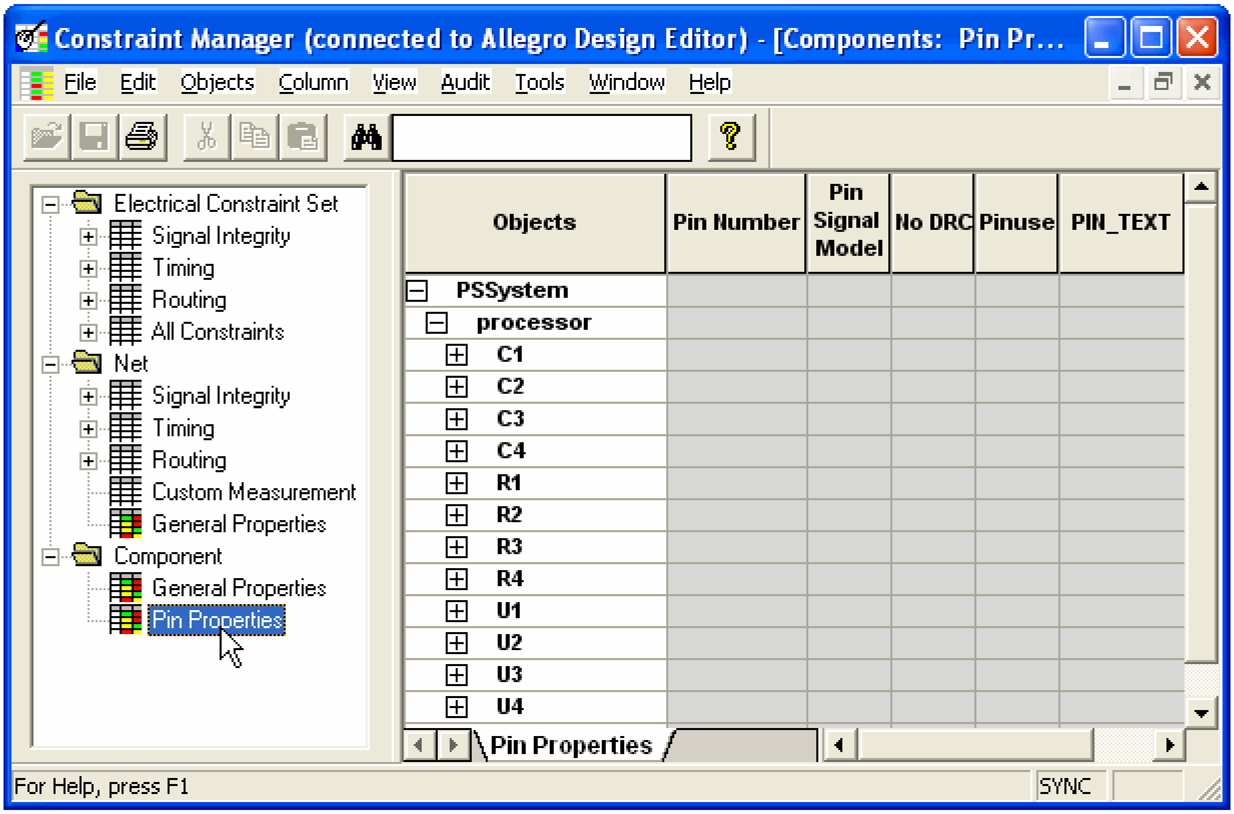


Note that the reference designators of components are displayed in Constraint Manager.

Note: To view the instance names of components, choose *Options* from the *View* menu in Constraint Manager, select the *Logical* option and click *OK*.

The changes you make to properties in Design Editor are displayed in Constraint Manager. Note that the `ROOM` property with the value `FPGA` you added on the `epf8282a` component using the Properties window in Design Editor is displayed in Constraint Manager. Similarly, the changes you make to properties in Constraint Manager are displayed in Design Editor.

The *Pin Properties* workbook in the *Component* object folder lets you work with properties on components pins.

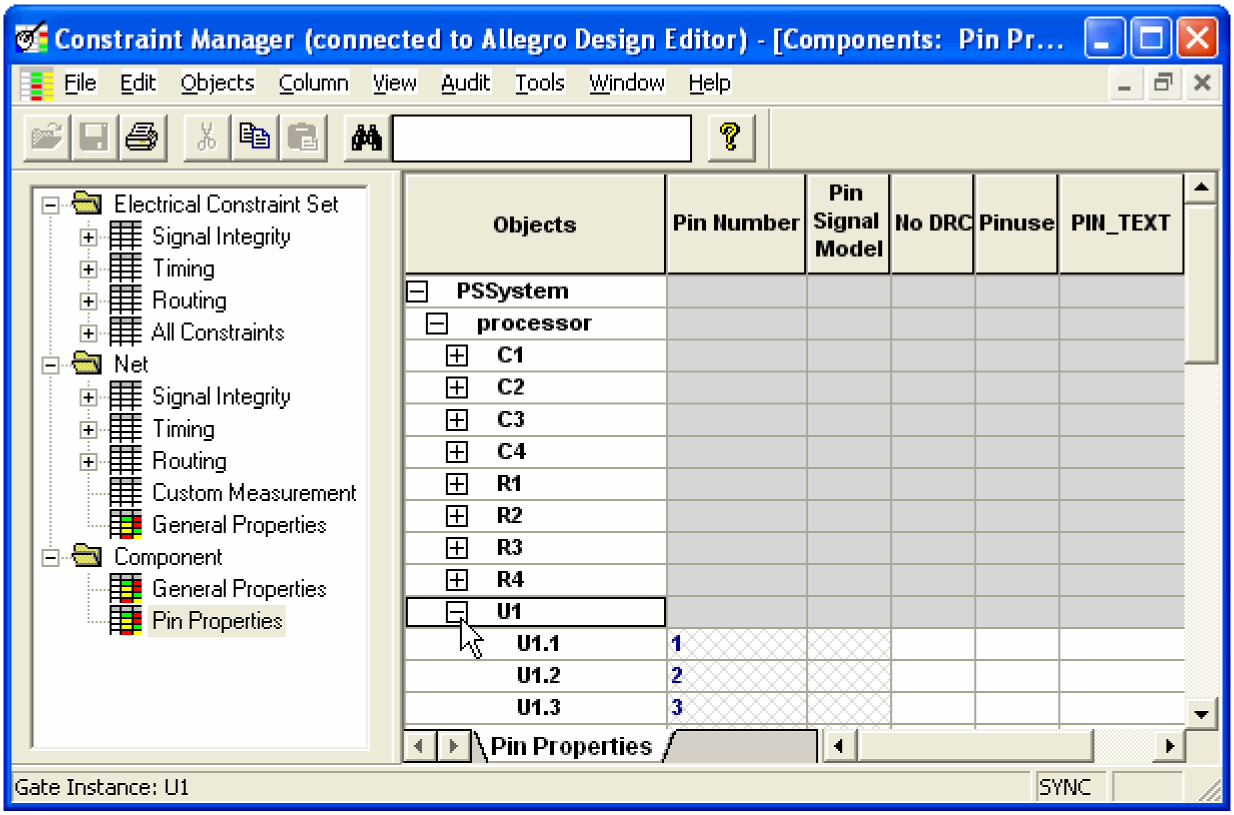


The reference designators of components are displayed in the *Pin Properties* worksheet.

To view the pins of a component, click the ⊞ icon next to the reference designator of the component.

2. Click the ⊞ icon next to the reference designator `U1`.

The pins on the component are displayed.

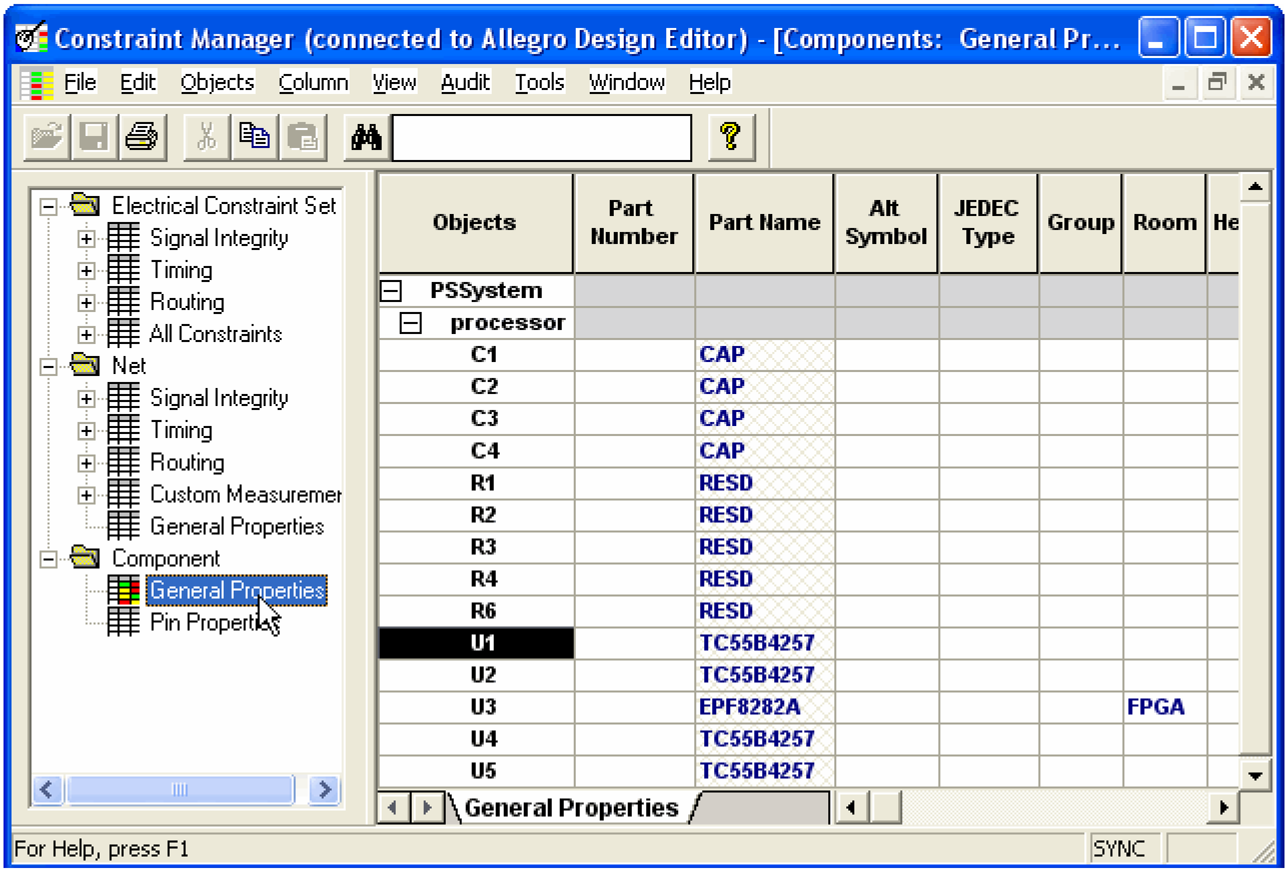


In the above figure, `U1.1` indicates the pin with the number `1` on the component with the reference designator `U1`.

Note: To view the pin names, choose *Options* from the *View* menu in Constraint Manager, select the *Logical* option and click *OK*.

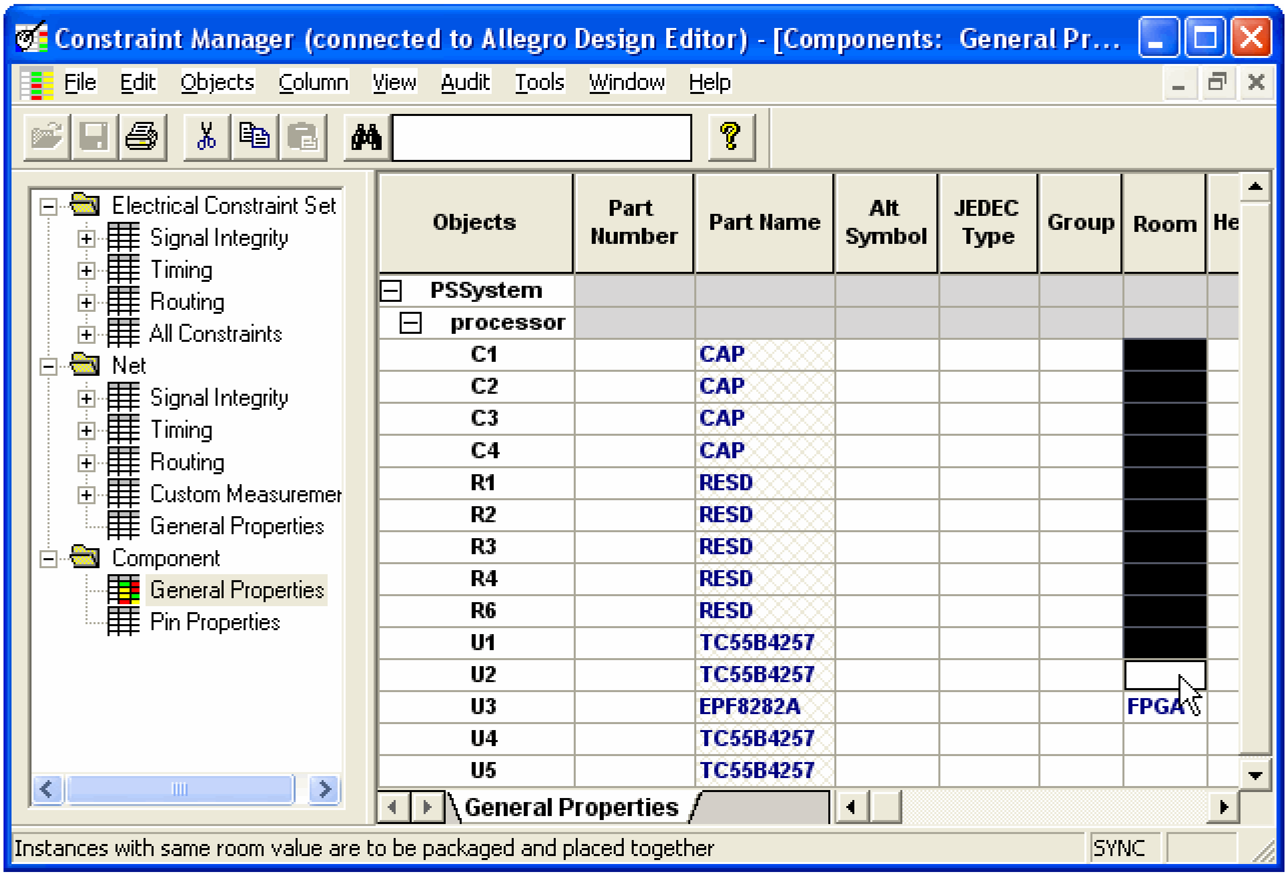
We will now add properties on components in Constraint Manager.

3. Click the *General Properties* workbook in the *Component* object folder.



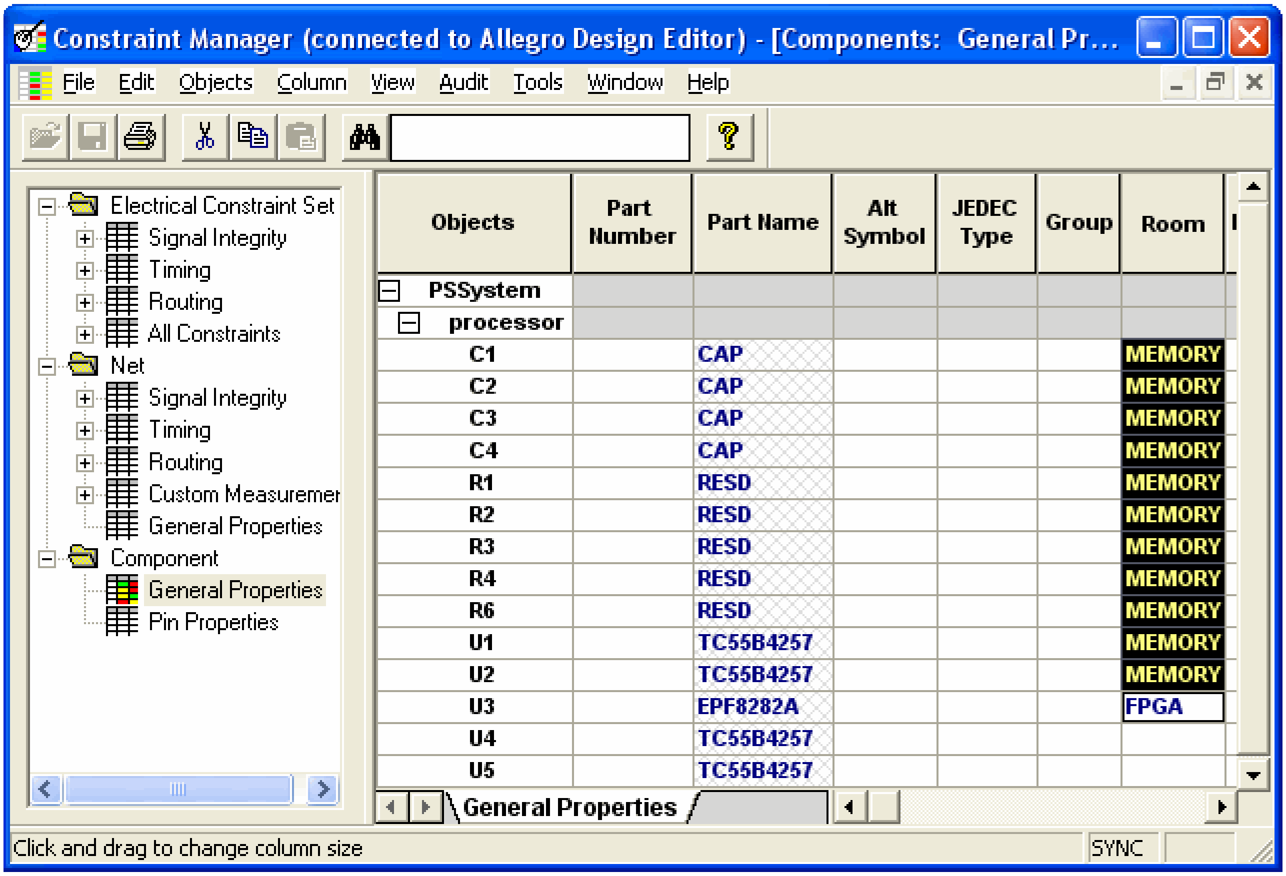
4. Click in the *Room* column next to the reference designator `C1`, press the *Shift* key, then click in the *Room* column next to the

reference designator U2, to select the cells in the *Room* column as shown below.

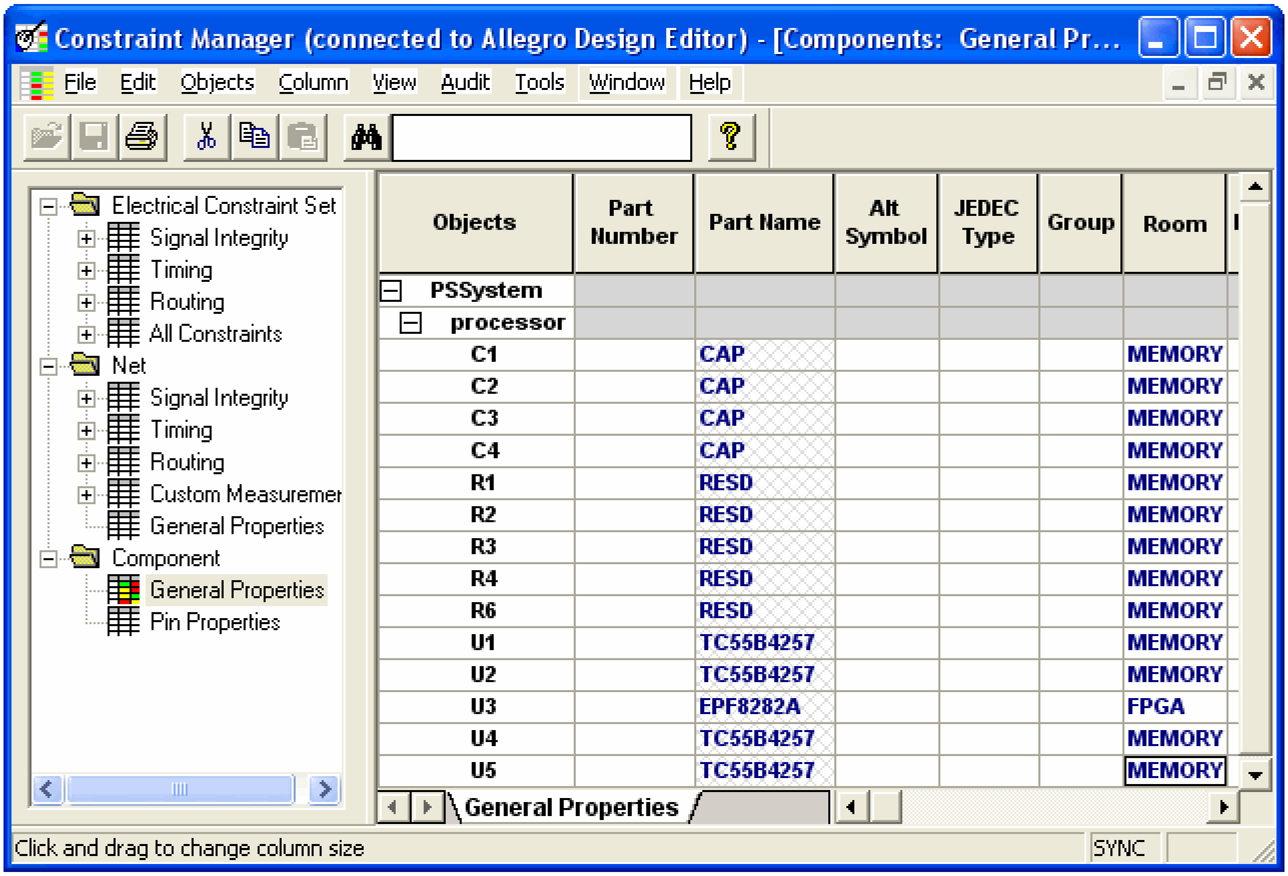


5. Type `MEMORY` and press *Enter*.

The `ROOM` property with the value `MEMORY` is added on the selected components. The `ROOM` property lets you control where parts are placed in the Allegro PCB Editor board.

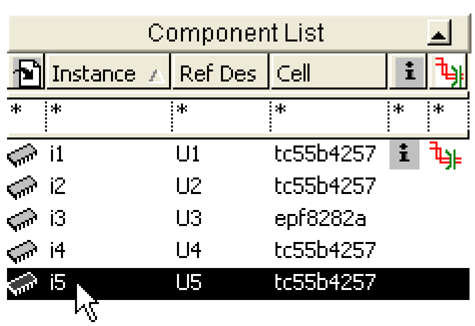


6. Add the `ROOM` property with the value `MEMORY` on the components with the reference designator `U4` and `U5`.

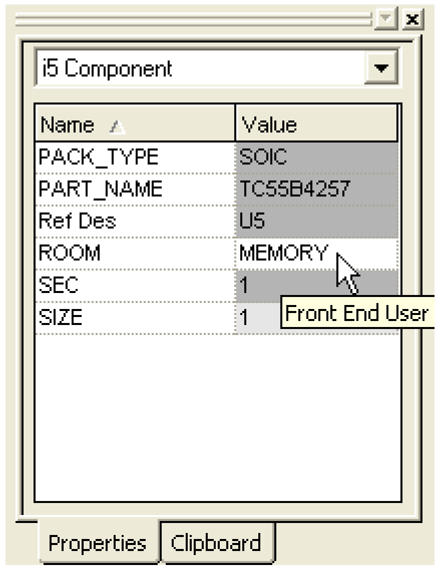


7. Switch to Design Editor.

8. In the Component List, select instance `i5` of the `tc55b4257` component.



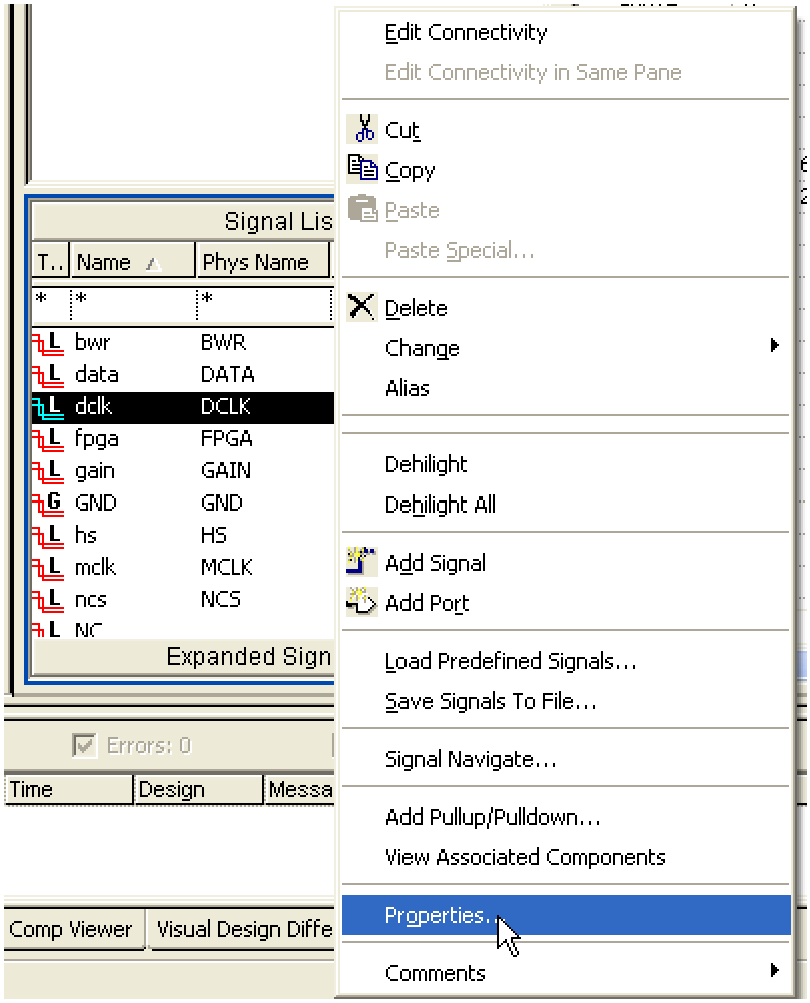
The Properties window displays the `ROOM` property you added on the component instance in Constraint Manager.



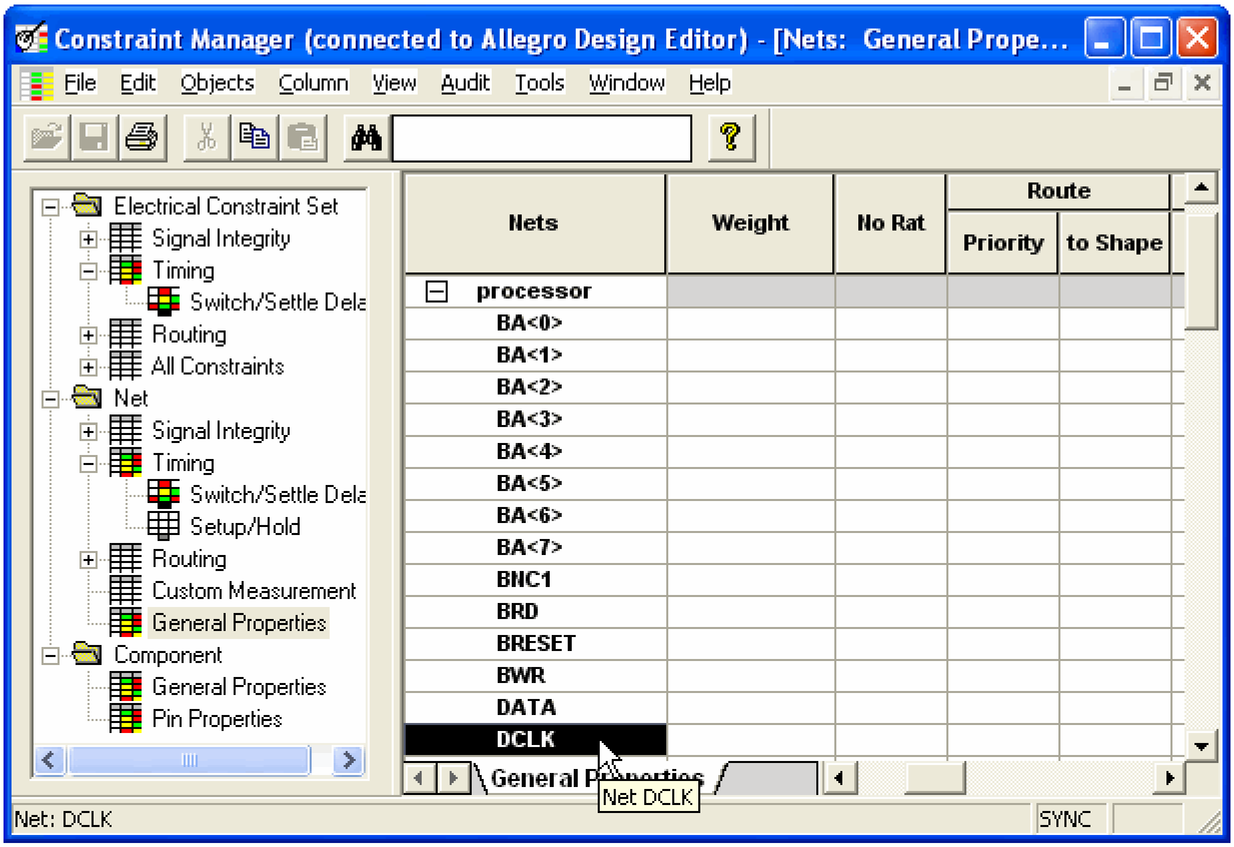
If you want to add a property on a specific component, net or pin in Constraint Manager, you can select the component, net or pin in Design Editor, click the right-mouse button and choose *Properties* to highlight the object in the corresponding property worksheet in Constraint Manager. Highlighting objects from Design Editor lets you quickly navigate to the required object in in Constraint Manager. This is especially useful when working with properties in large designs where there are hundreds of components and nets in the design and when you are working with properties on pins of large pin-count devices.

9. In the Signal List in Design Editor, select the `dclk` signal.

10. Click the right-mouse button and choose *Properties* from the shortcut menu.



The net `DCLK` is highlighted in the *General Properties* workbook in the *Net* object folder.



This allows you to quickly add properties on specific components, nets and pins in the design.

Summary

You now know how to use Constraint Manager to work with properties on components, nets and pins in the design.

For More Information

See:

- Working with Design Editor Properties chapter of the Allegro Constraint Manager User Guide.

- Working with Properties and Electrical Constraints chapter of *Allegro Design Editor User Guide*.

Lesson 4-3: Defining User Defined Properties

Overview

You can use an user defined property to capture a characteristic of an object. Design Editor and Constraint Manager do not perform any design rule checks or analysis on user defined properties; they facilitate communication of the design intent to down-stream tools in which you may want to manipulate the design objects associated with these properties.

Design Editor lets you define user defined properties. You can also define user defined properties in Constraint Manager.

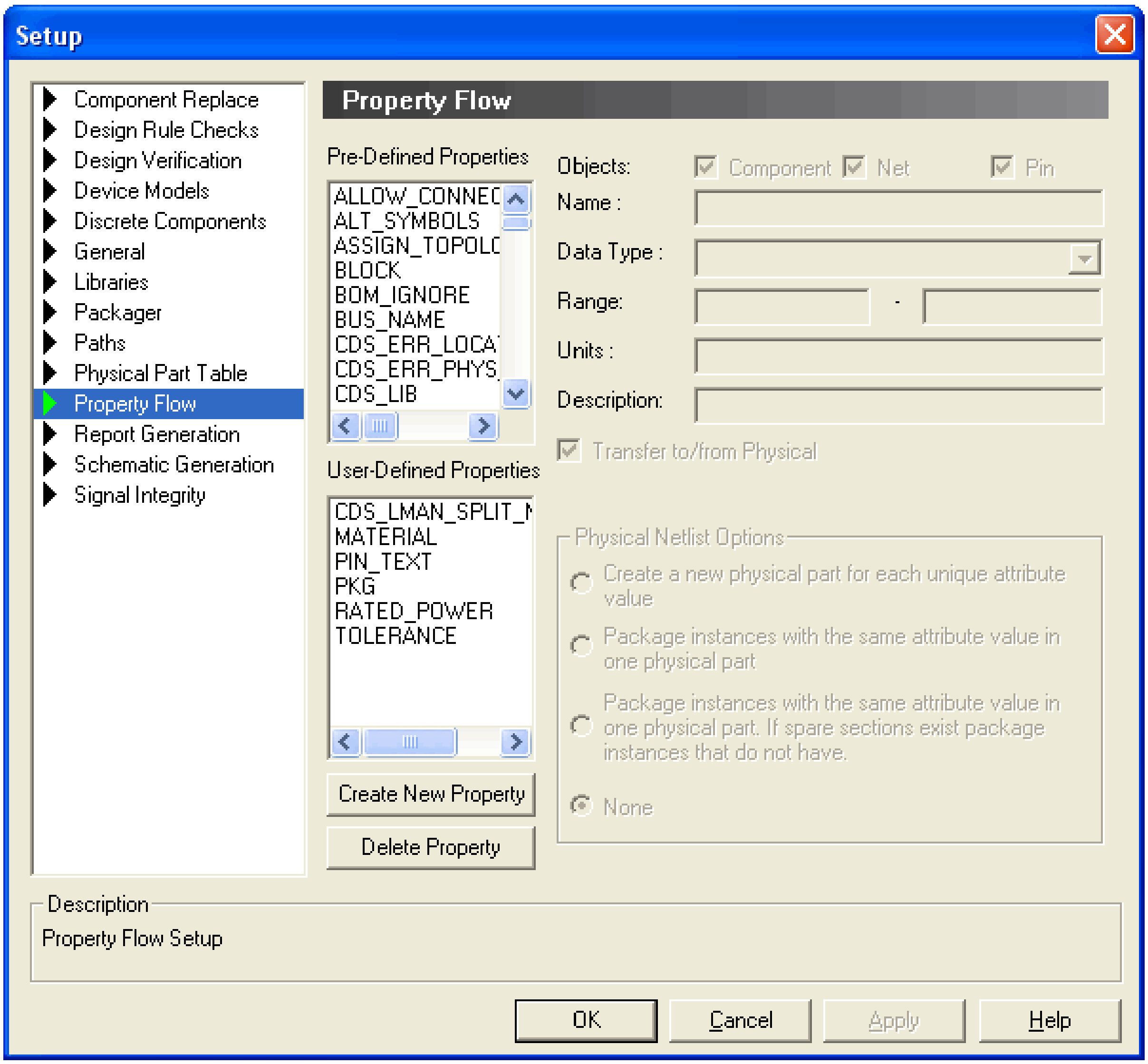
In this lesson you will learn to add a user defined property in Design Editor.

Procedure

1. In Design Editor, choose *Project – Settings*.

   The Setup dialog box appears.

2. Click the *Property Flow* tab.



The predefined set of properties supported by Design Editor are displayed in the *Pre-Defined Properties* list.

The user defined properties are displayed in the *User-Defined Properties* list.

3. Click the *Create New Property* button.

4. In the *Name* field, enter the property name as:

`COST`

5. Deselect the *Net* and *Pin* check boxes.

   As only the *Component* check box is selected, you can add the `COST` property only on components in the design.

6. From the *Data Type* drop-down list, choose *String*.

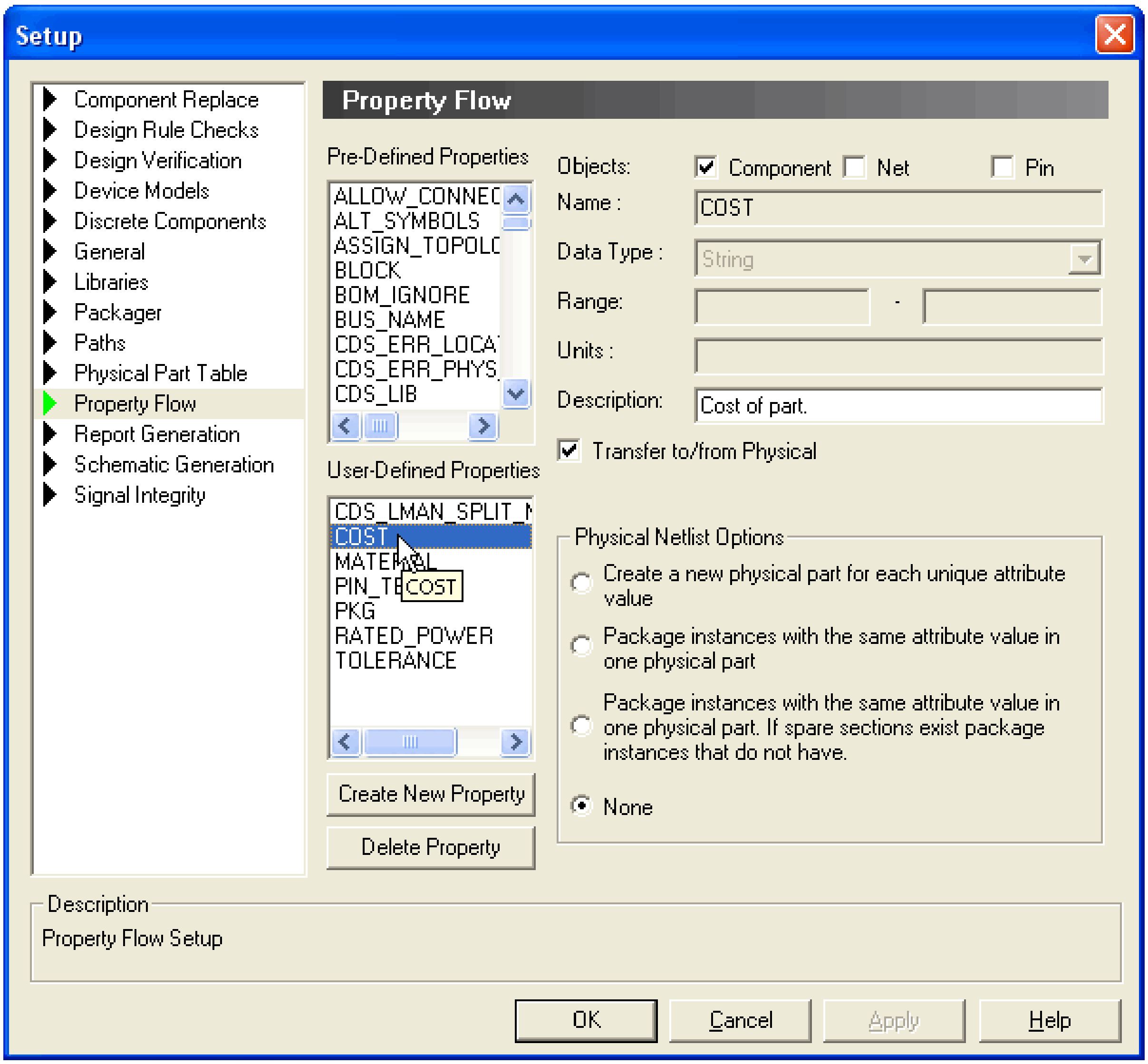
7. In the *Description* field, enter:

   `Cost of part.`

8. Click *Apply*.

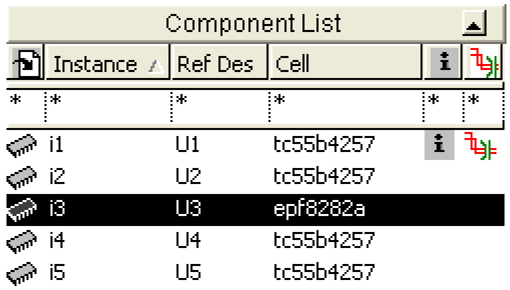
   The `COST` property is displayed in the *User-Defined Properties* list.

9. Click on the `COST` property to view its definition.



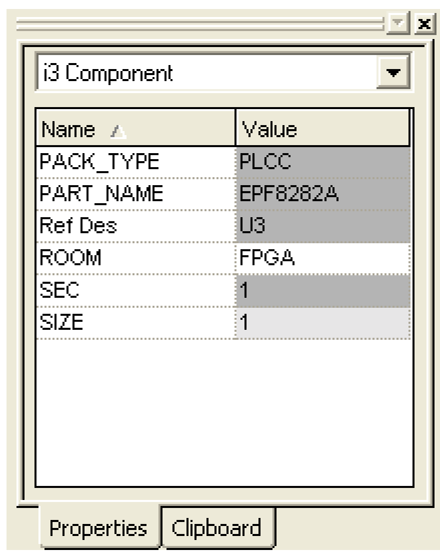
You will now add the `COST` property on the `epf8282a` component in the design.

10. In the Component List, select the `epf8282a` component.



11. Choose *View – Properties Window*.

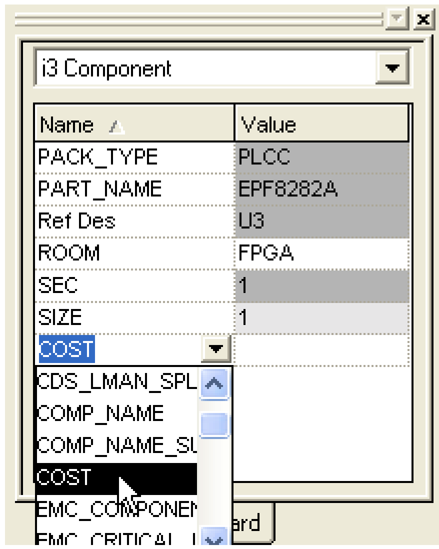
The Properties window displays the properties on the `epf8282a` component.



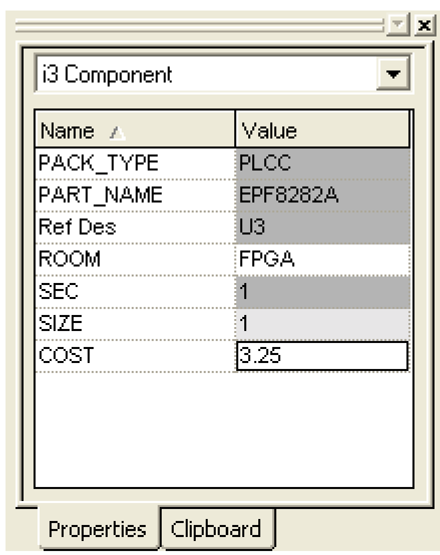
12. Click in the Properties window and press the *Insert* key.

A new row is added in the Properties window.

13. Click in the *Name* column in the new row and choose `COST` from the drop-down list.



14. In the *Value* column, enter `3.25` as the value for the `COST` property.



Summary

You now know how to define user defined properties in Design Editor and add user defined properties on components, nets and pins in the design.

For More Information

See:

- Working with Properties and Electrical Constraints chapter of *Allegro Design Editor User Guide* for more information on working with user defined properties in Design Editor.
- Working with Design Editor Properties chapter of the Allegro Constraint Manager User Guide for more information on working with user defined properties in Constraint Manager.

Module 5: Working with Electrical Constraints

Prerequisite

If you have not completed all the lessons in the previous modules, you must open the `tutorial.cpm` project located at `<your_work_area>\modules\constraints\tutorial` in Design Editor and perform the steps described in this module.

For more information, see Understanding the Sample Design Files on page 14.

Lessons

This module consists of the following lessons:

Completion Time

1 hour for written lessons

Overview

Allegro Constraint Manager is a workbook- and worksheet-based application used to manage high-speed constraints across all tools in the Cadence PCB and IC Package design flow.

You can use Constraint Manager with Design Editor to capture electrical constraint information early in the design cycle—right at the logic design stage.

Constraint Manager lets you define, view, and validate constraints at each step in the design flow, from design capture (in Allegro Design Entry HDL or Allegro Design Editor) to floorplanning (in Allegro PCB SI to design realization (in Allegro PCB). You can also use Constraint Manager with SigXplorer to explore circuit topologies and derive electrical constraint sets which can include custom constraints, custom measurements, and custom stimulus.

In this module you will learn to use Constraint Manager with Design Editor.

Lesson 5-1: Starting Constraint Manager from Design Editor

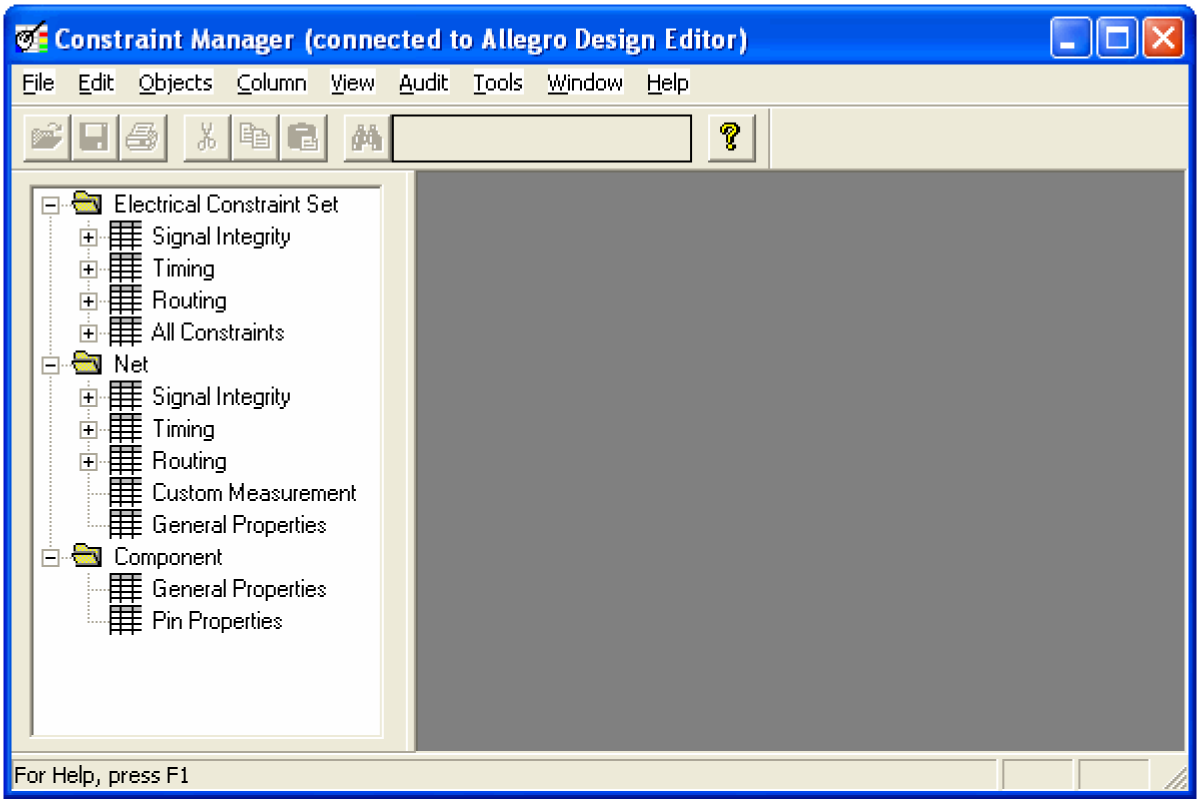
Overview

In this lesson, you will learn to start Constraint Manager from Design Editor.

Procedure

1. To start Constraint Manager, do one of the following:
   - ❑ Click the  toolbar button.
   - ❑ Choose *Design – Edit Constraints*.

The Constraint Manager window appears.



The Constraint Manager title bar displays (connected to Allegro Design Editor). This indicates that Constraint Manager has been started from Design Editor.

Note: If you want to use Constraint Manager with Design Editor, you must start Constraint Manager only from Design Editor. If you start Constraint Manager from Allegro PCB Editor, you cannot use Constraint Manager to manage constraints in Design Editor.

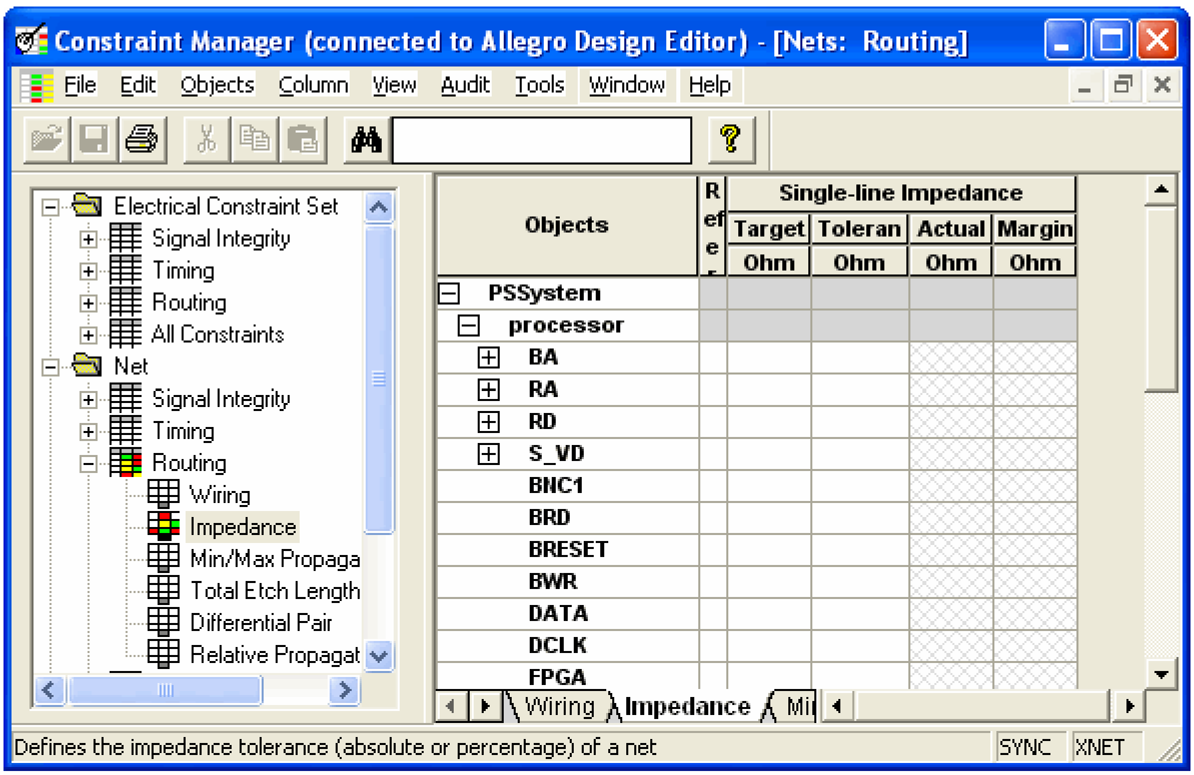
The Constraint Manager user interface has two windows or panes. The left pane is the worksheet selector window. It contains three main folders.

- The *Electrical Constraint Set* folder is used to create constraint rule sets. Think of this folder as a storage area of predefined constraint rules.
- The *Net* folder is used to assign individual constraint rules (or predefined rule sets) to nets. You can also attach constraint rules

to pin pairs, buses, differential pairs, as so on. You can use the *General Properties* workbook in the *Net* folder to assign properties on nets.

- The *Component* folder is used to assign properties on components and pins.

The right pane is the worksheet editor window.



You can expand each folder in the left pane to view their associated workbooks. Each workbook contains one or more worksheets. The selected worksheet is displayed in the right pane. Each workbook deals with a particular category of design rules: Signal Integrity, Timing, and Routing. To attach routing rules to nets in the design, expand the *Net* folder, and edit the worksheets stored inside the Routing workbook.

Summary

You now know how to start Constraint Manager from Design Editor. You also learned about the folders, workbooks and worksheets in Constraint Manager.

For More Information

See:

Allegro Constraint Manager Reference for more information on each worksheet in Constraint Manager.

Lesson 5-2: Assigning Constraints on a Net

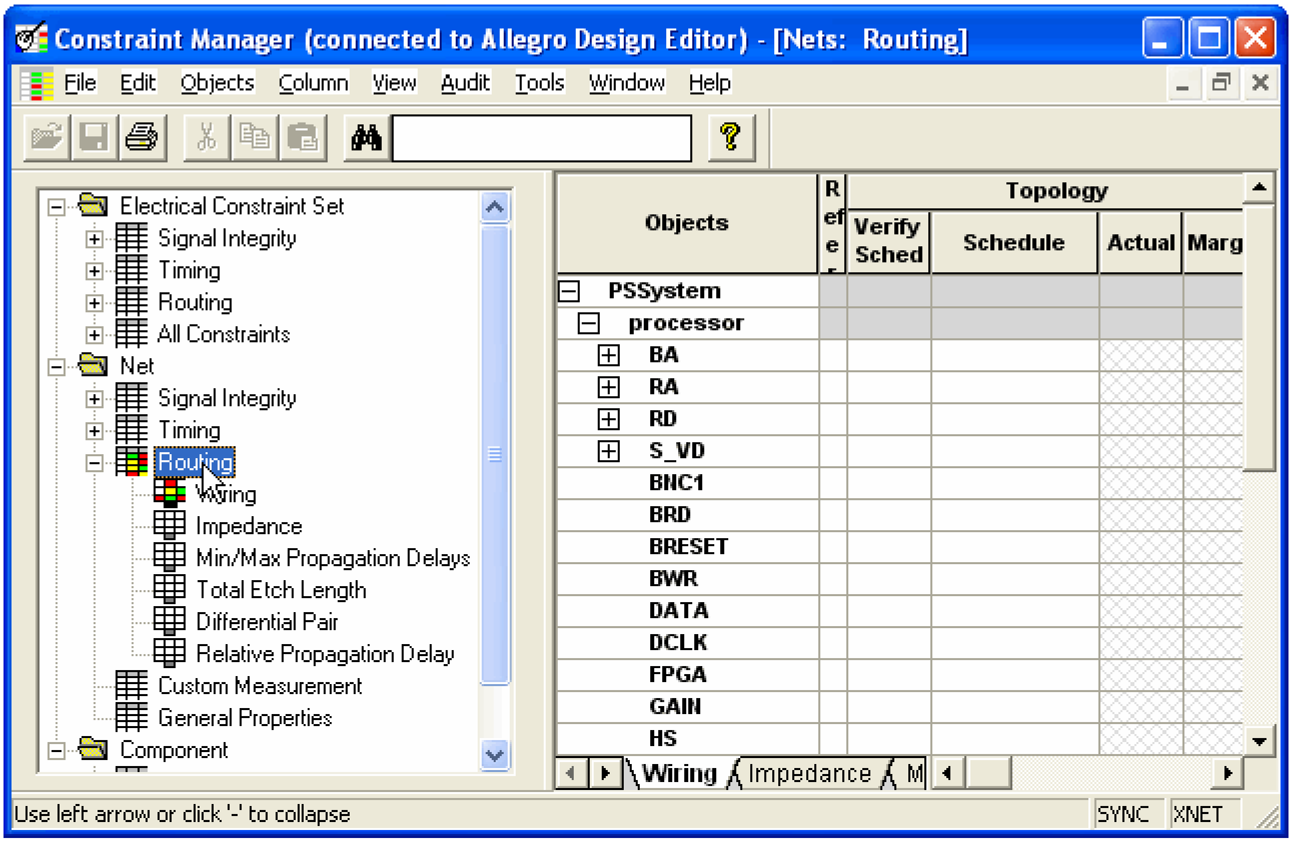
Overview

While designing the schematic for your design, you might have several design constraints such as length and impedance on the critical nets in the design. These constraints might have been given to you by the Signal Integrity engineer. These translate to the length of critical nets and therefore to the propagation delay of the signals passing through them.

In this lesson you will learn to set the minimum and maximum propagation delay constraint for a net.

Procedure

1. In the *Net* folder, double-click on the *Routing* workbook.



The worksheets in the *Routing* workbook are displayed. Note that nets are displayed using physical net names in Constraint Manager.

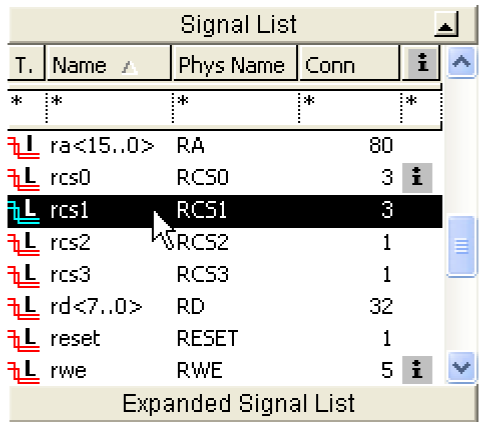
2. Click on the *Min/Max Propagation Delays* worksheet.

You will now set the minimum and maximum propagation delay constraint for the net named RCS1. First, you will set the delay between all the drivers and receivers of net RCS1. Then, for a specific driver and receiver pair, we will set a different delay value.

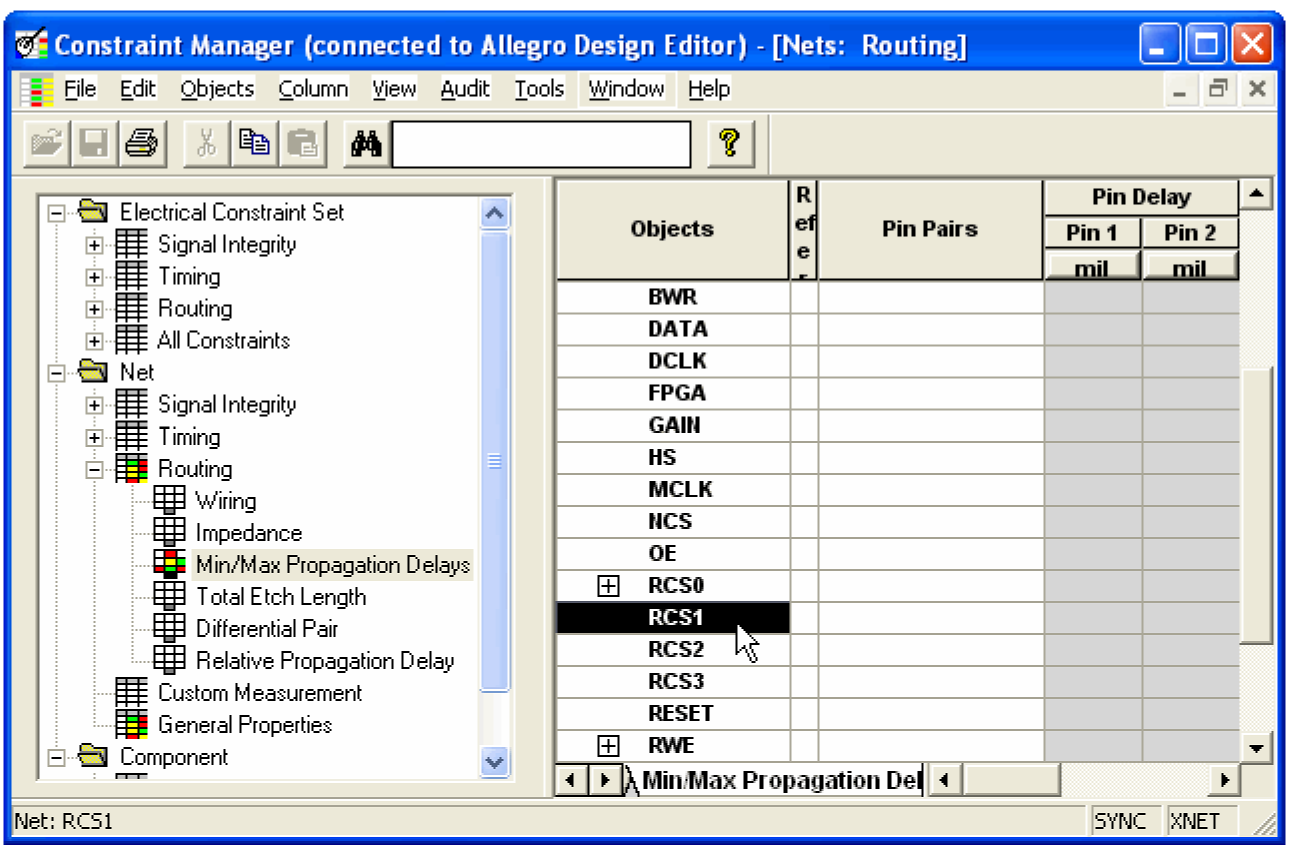
You can locate the RCS1 net in the *Min/Max Propagation Delays* worksheet or select the net in the Signal List in Design Editor to highlight the net in the *Min/Max Propagation Delays* worksheet.

3. In the Signal List in Design Editor, select the `rcs1` net.

   Note that the net `rcs1` has the physical net name `RCS1`. This physical net name is displayed in Constraint Manager.



   See that the net is highlighted in the *Min/Max Propagation Delays* worksheet in Constraint Manager.



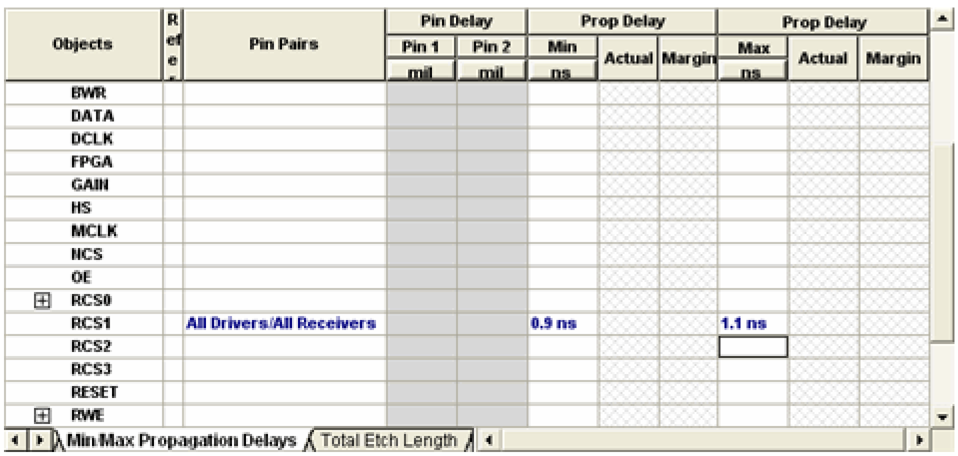
Highlighting nets from Design Editor lets you quickly navigate to the required net in the selected worksheet in Constraint Manager. This is especially useful in large designs where there are hundreds of nets in the design.

4. In the *Prop Delay* column, enter the minimum propagation delay for the net `RCS1` as:

   `0.9`

5. In the *Prop Delay* column, enter the maximum propagation delay for the net `RCS1` as:

   `1.1`



Objects	Ref	Pin Pairs	Pin Delay Pin 1 mil	Pin Delay Pin 2 mil	Prop Delay Min ns	Actual	Margin	Prop Delay Max ns	Actual	Margin
BWR										
DATA										
DCLK										
FPGA										
GAIN										
HS										
MCLK										
NCS										
OE										
RCS0										
RCS1		All Drivers/All Receivers			0.9 ns			1.1 ns		
RCS2										
RCS3										
RESET										
RWE										

The default unit of value for the propagation delay constraint is `ns` (nanoseconds). This means that the signal on the net `RCS1` must have a propagation delay of atleast `0.9 ns` before it reaches any destination, and that the signal must reach any destination within `1.1 ns` after it is available on the net `RCS1`.

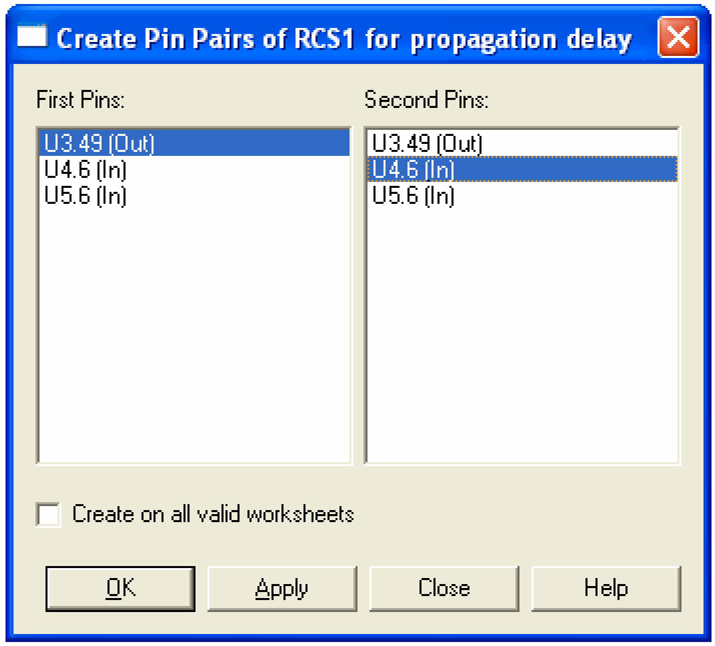
Note that in the *Pin Pairs* column, *All Drivers/All Receivers* gets selected automatically. This means that the propagation delay has been set between all the drivers and receivers of the signal on the net `RCS1`.

6. Click net `RCS1` and choose *Objects – Create – Pin Pair*.

   The Create Pin Pairs of RCS1 for propagation delay dialog box appears. The *First Pins* column lists the driver pins connected to the net `RCS1`. The *Second Pins* column list the receiver pins connected to the net `RCS1`.

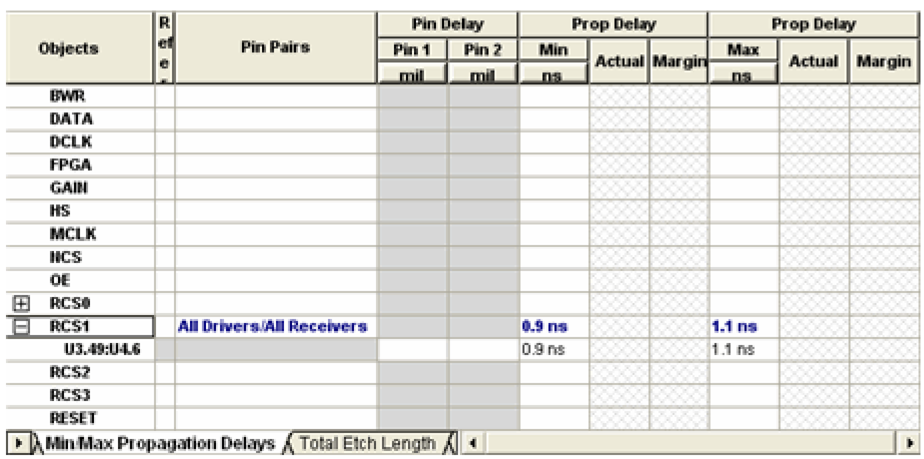
7. In the *First Pins* column, click `U3.49`.

8. In the *Second Pins* column, click `U4.6`.



9. Click *OK* to create the pin pair.

The pin pair `U3.49:U4.6` is displayed under the net `RCS1` in the Constraint Manager window.

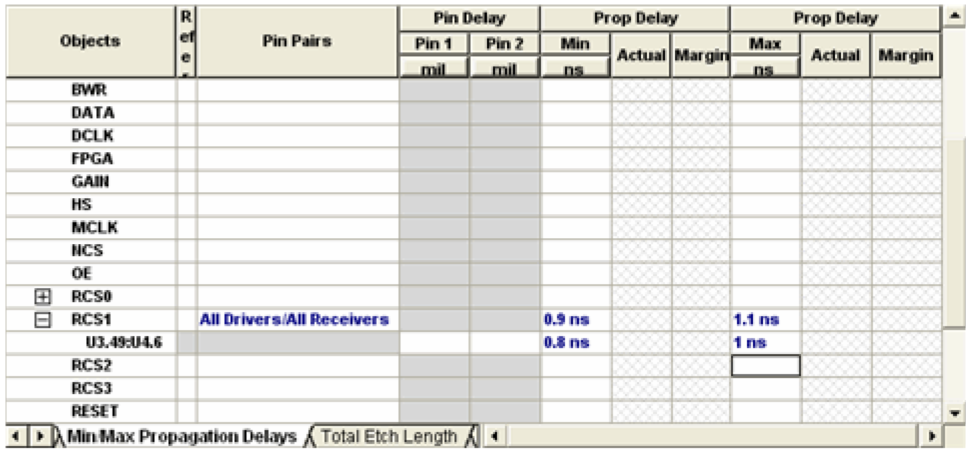


The propagation delay constraint on the `RCS1` net is inherited by the pin pair. You can override the inherited constraint values.

10. Change the value in the *Min* column for the pin pair from `0.9 ns` to `0.8 ns`.

11. Change the value in the *Max* column for the pin pair from `1.1 ns` to `1.0 ns`.

The worksheet appears as shown below:



Objects	Ref	Pin Pairs	Pin Delay Pin 1 (mil)	Pin Delay Pin 2 (mil)	Prop Delay Min (ns)	Actual	Margin	Prop Delay Max (ns)	Actual	Margin
BWR										
DATA										
DCLK										
FPGA										
GAIN										
HS										
MCLK										
NCS										
OE										
⊞ RCS0										
⊟ RCS1		All Drivers/All Receivers			0.9 ns			1.1 ns		
U3.49:U4.6					0.8 ns			1 ns		
RCS2										
RCS3										
RESET										

Summary

You now know how to assign constraints on nets. You also learned to create a pin pair and override the constraint inherited by the pin pair from a net.

For More Information

See:

Allegro Constraint Manager User Guide for more information on using Constraint Manager.

Lesson 5-2: Working with Electrical Constraint Sets

Overview

You can identify the critical nets in your design and then identify constraints that are applicable to all of them. You can then define those constraints together in an ECSet and assign the ECSet on each of the critical nets. Thus, an ECSet can be used to define a generic set of rules applicable to a number of nets. If your design requirement changes at a later point in time, you can edit the constraints on the ECSet; all the nets referencing the ECSet shall inherit the changed ECSet automatically. Thus, using ECSets is a very efficient way of capturing constraints.

Another use of ECSets is in the case of design reuse. If you are reusing a design that has ECSets defined for its critical nets, you can import the ECSets into your new design; this saves a lot of rework.

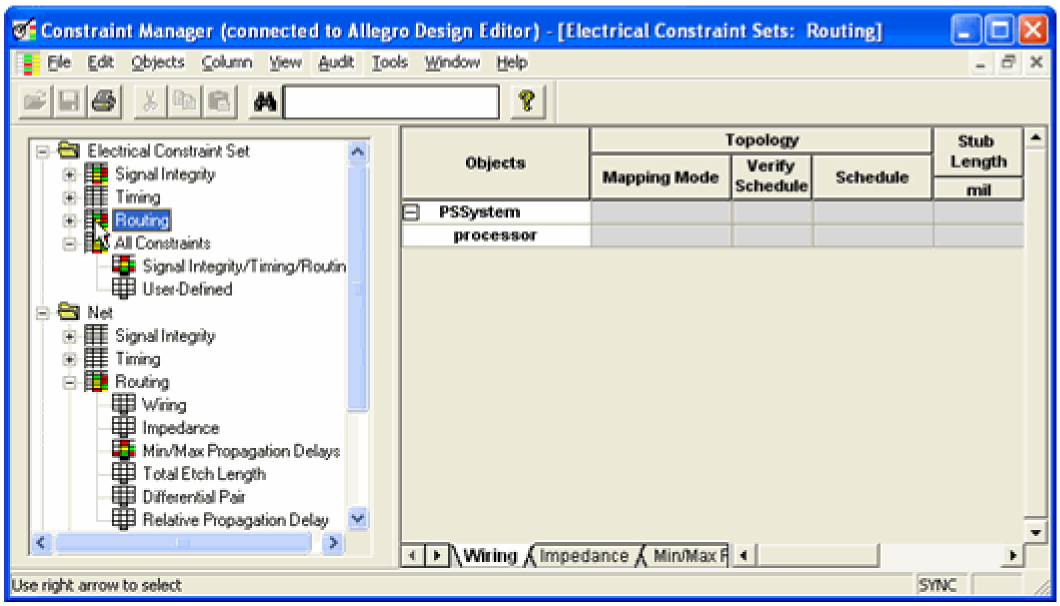
The main advantages of an ECSet are:

- The ECSet can be assigned to many nets simultaneously.
- You can capture any or all electrical constraints in one ECSet.
- A change in a constraint in an ECSet is automatically inherited by the objects that reference the ECSet.
- You can override the constraints inherited from an ECSet.
- You can assign a different ECSet if the requirements change.

In this lesson, you will create an ECSet named `CRITICAL` and assign it on the bus `S_VD`. You will then override the constraints inherited from the ECSet on a bit of the bus.

Procedure

1. In the Constraint Manager window, click the *Routing* workbook in the *Electrical Constraint Set* folder.



2. Choose *Objects – Create – Electrical CSet*.

   The Create Electrical CSet dialog box appears.

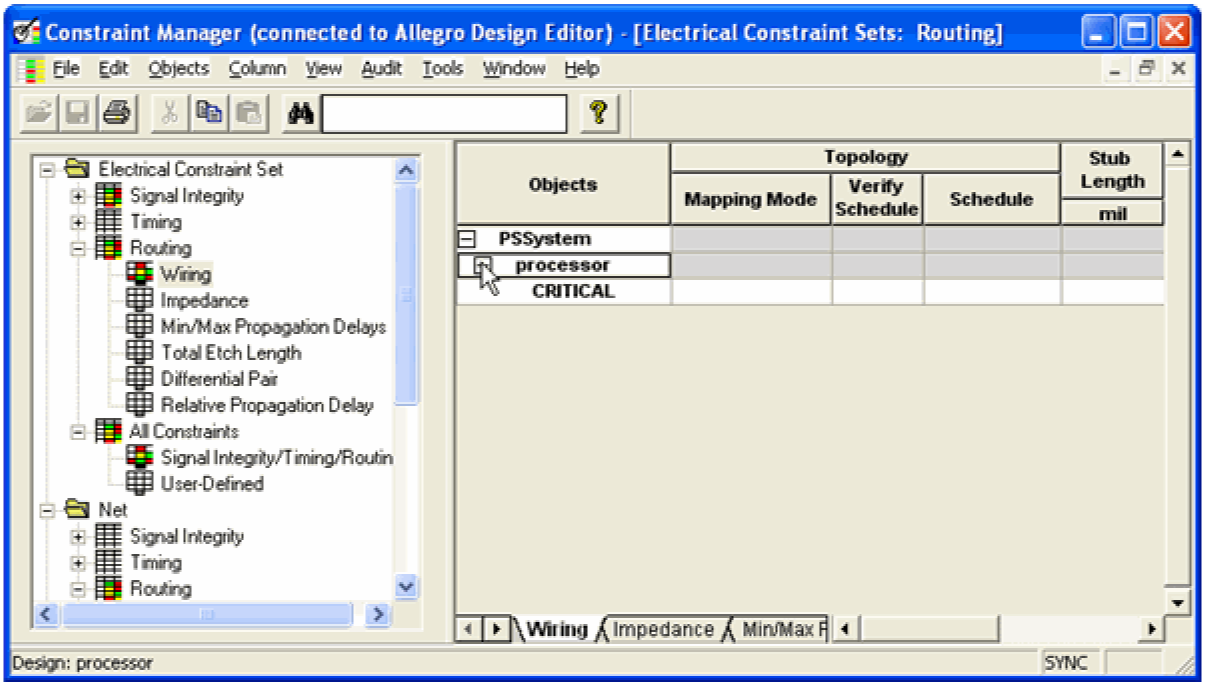
3. Enter the ECSet name as:

   `CRITICAL`

4. Click *OK*.

5. In the *Objects* column, click on the ⊞ sign next to the `processor` design.

The design expands to show the name of the new ECSet.



6. Click the *Wiring* worksheet in the *Routing* workbook in the *Electrical Constraint Set* folder.

   You will now specify the wiring rules for the ECSet.

7. Do the following:

   a. Click in the *Verify Schedule* field, and select *Yes*.

   b. Click in the *Schedule* field and select *Source-load Daisy-chain*.

   c. Click in the *Stub Length* field and enter `150`.

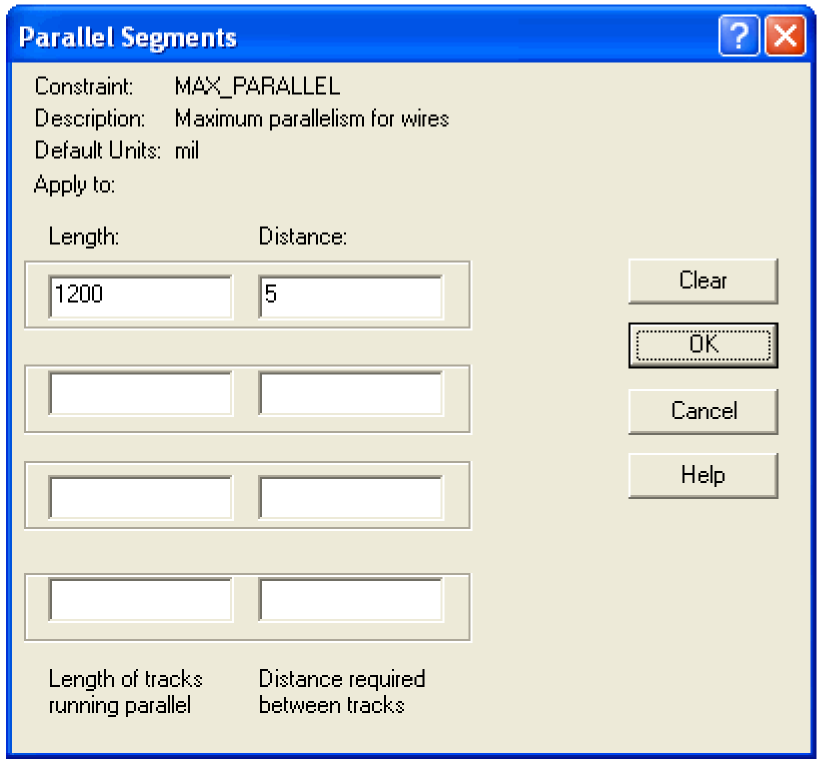
   d. Click in the *Max Via Count* field and enter `4`.

   e. Click in the *Max Parallel* field.

      The Parallel Segments dialog box appears.

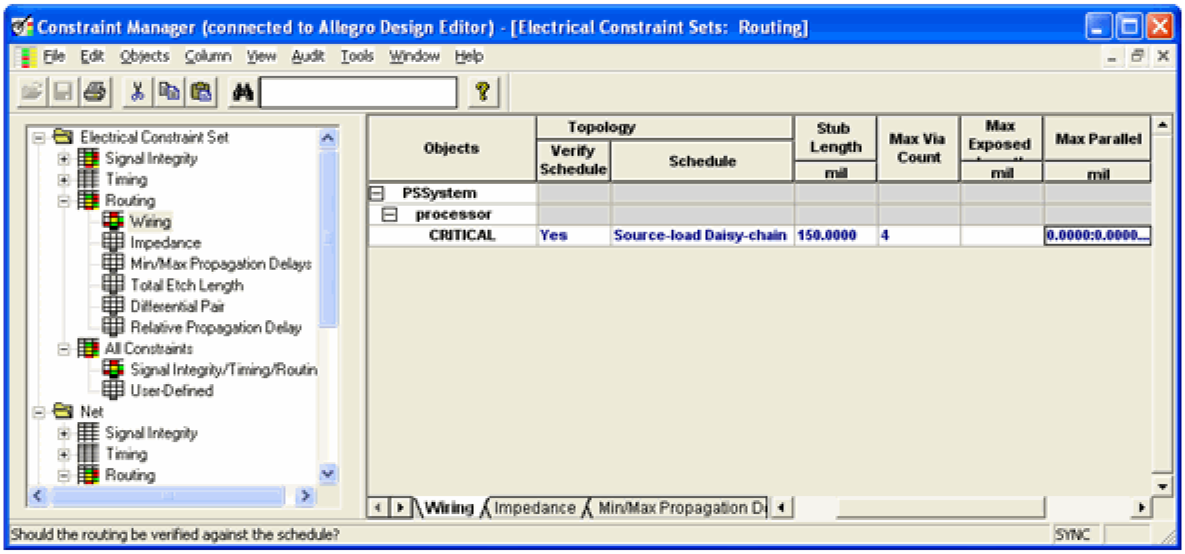
   f. Click in the first *Length* field and type `1200`.

g. Click in the *Distance* field next to the first *Length* field and type 5.



h. Click OK.

The *Wiring* worksheet appears as shown below:



8. Click the *Impedance* worksheet in the *Routing* workbook in the *Electrical Constraint Set* folder.

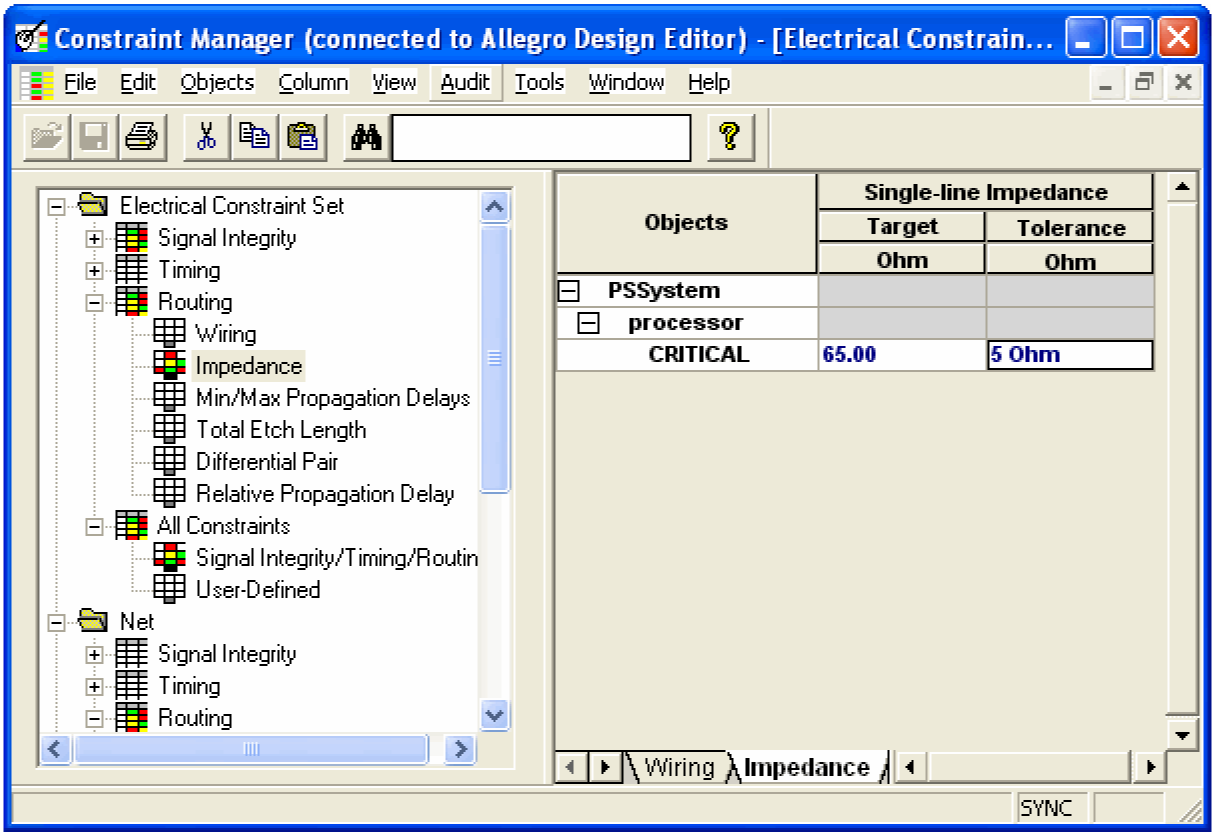
   You will now specify the impedance rules for the ECSet.

9. Do the following:

   a. Click in the *Target* field and enter 65.

   b. Click in the *Tolerance* field and enter 5.

The *Impedance* worksheet appears as shown below:



10. Click the *Min/Max Propagation Delay* worksheet in the *Routing* workbook in the *Electrical Constraint Set* folder.

You will now specify the delay rules for the ECSet.

11. Do the following:

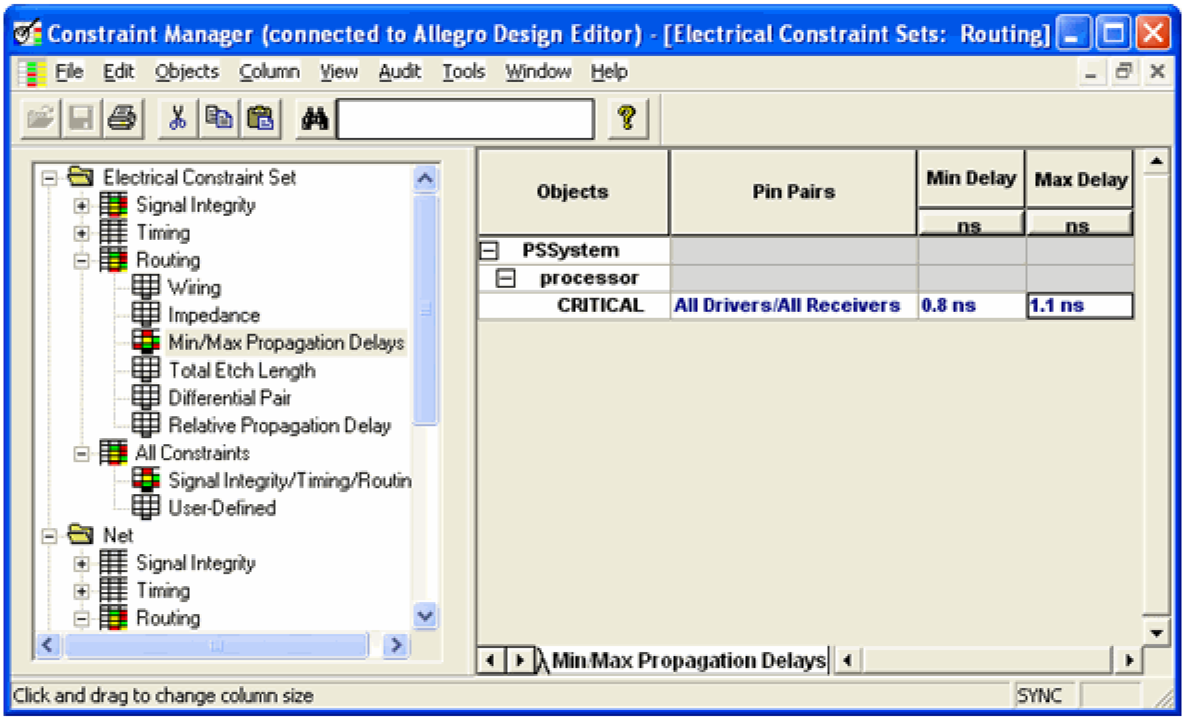
a. Click in the *Min Delay* field and enter:

```
0.8
```

b. In the *Max Delay* field, enter:

```
1.2
```

The *Impedance* worksheet appears as shown below:



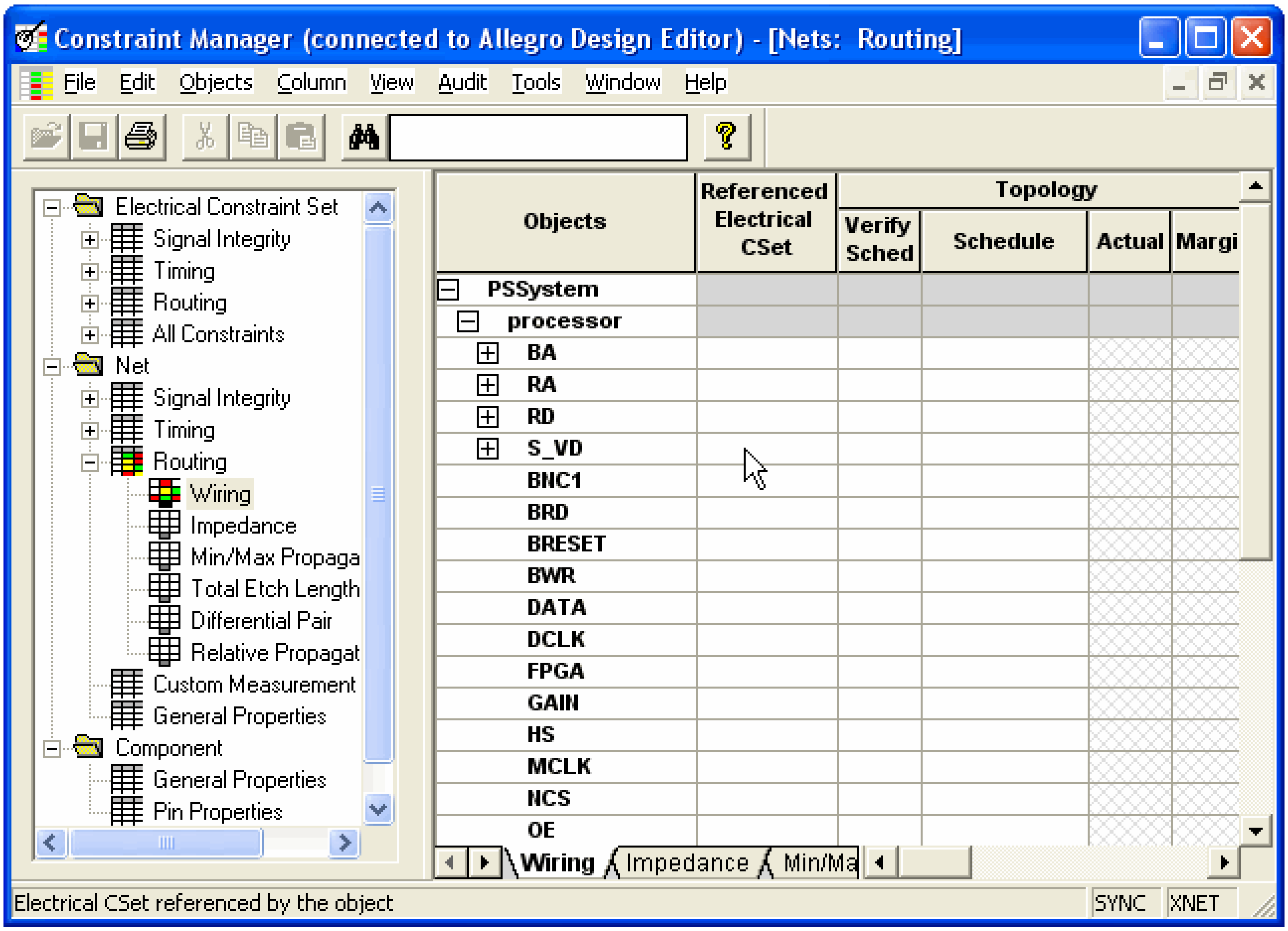
You used the *Routing* workbook in the *Electrical Constraint Set* folder to define the following rules:

- A source-load daisy chain schedule - a special routing sequence based on pin type.
- A stub length rule - 150 mils.
- A maximum vias per net rule - 4 maximum.
- A parallelism rule - two nets 5 mils apart cannot be adjacent more than 1200 mils.
- An impedance rule - 65 ohms.
- A min/max delay rule - the time taken for a signal to travel from the driver to the closest receiver must be at least 0.8 nanoseconds, and the time taken for the signal to travel to the furthest receiver not more than 1.2 nanoseconds.

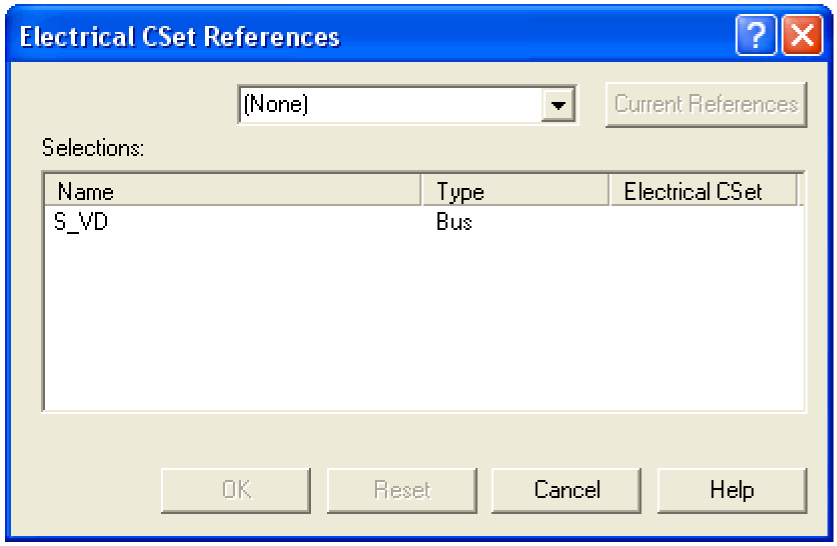
These rules are stored in the ECSet named `CRITICAL`. You will now assign the ECSet named `CRITICAL` on the bus `S_VD`.

12. Click the *Wiring* worksheet in the *Routing* workbook in the *Net* folder.

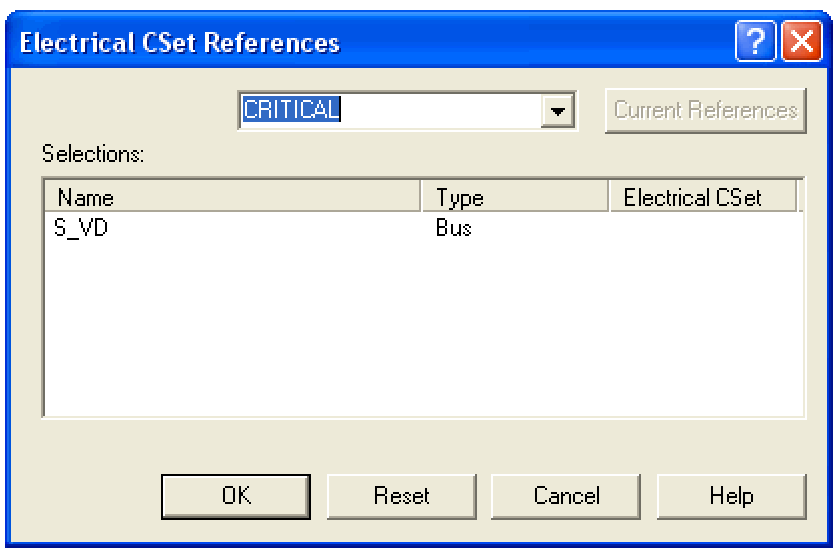
13. Double-click on the Referenced Electrical CSet column next to the bus S_VD.



The Electrical CSet References dialog box appears.

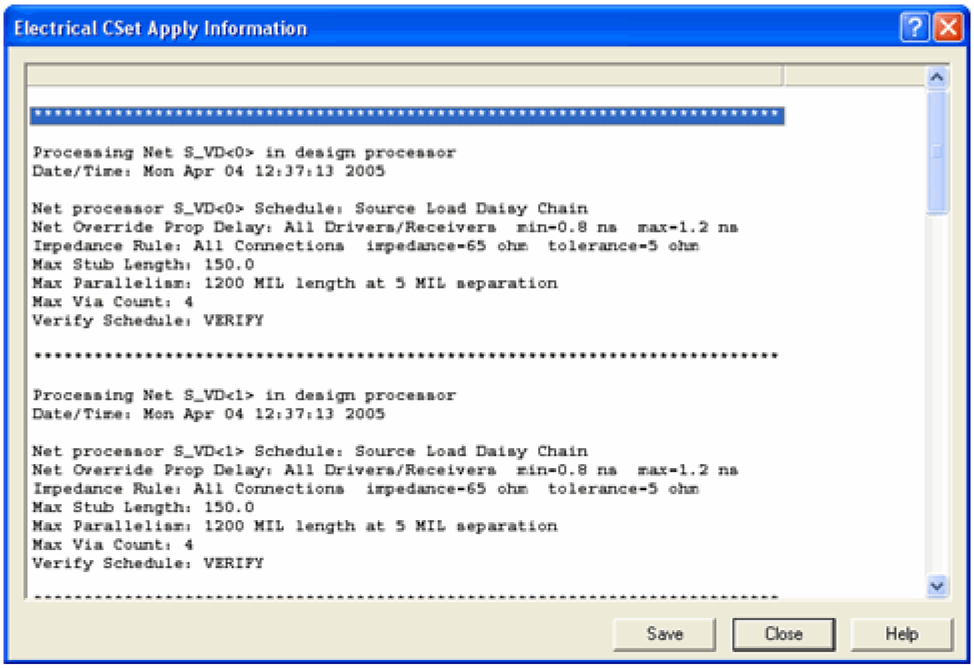


14. Click the drop-down list and choose *CRITICAL*.



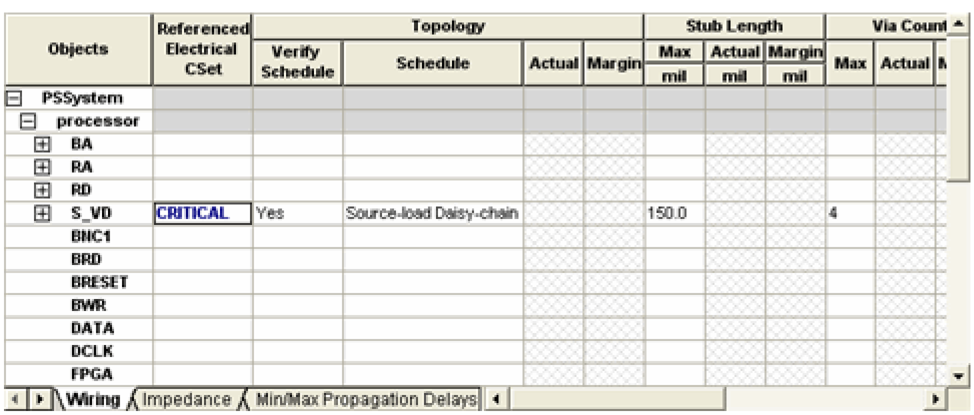
15. Click *OK*.

The Electrical CSet Apply Information dialog box appears.



16. Click *Close*.

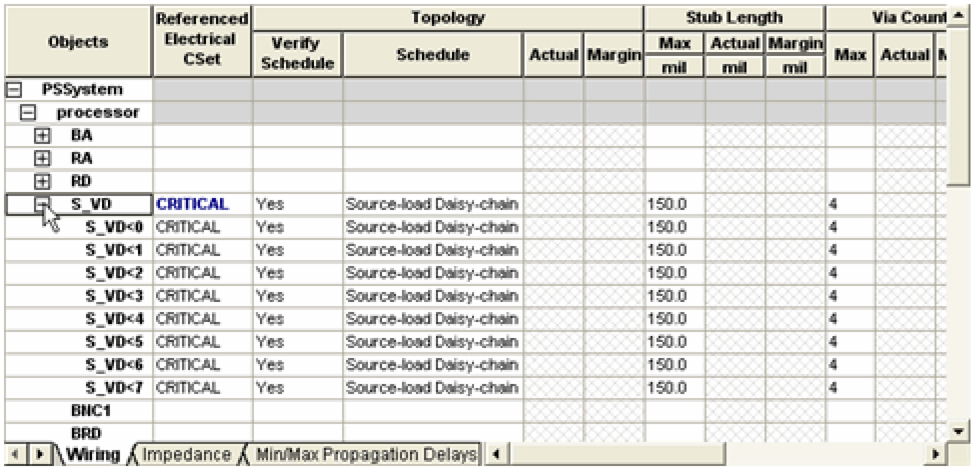
The ECSet is assigned on the bus and the constraints on the ECSet are displayed on the bus. The constraints are displayed in gray color because they are inherited constraints.



Objects	Referenced Electrical CSet	Topology				Stub Length			Via Count	
		Verify Schedule	Schedule	Actual	Margin	Max mil	Actual mil	Margin mil	Max	Actual
PSSystem										
processor										
BA										
RA										
RD										
S_VD	CRITICAL	Yes	Source-load Daisy-chain			150.0			4	
BNC1										
BRD										
BRESET										
BWR										
DATA										
DCLK										
FPGA										

17. Click on the ⊞ sign next to the bus S_VD.

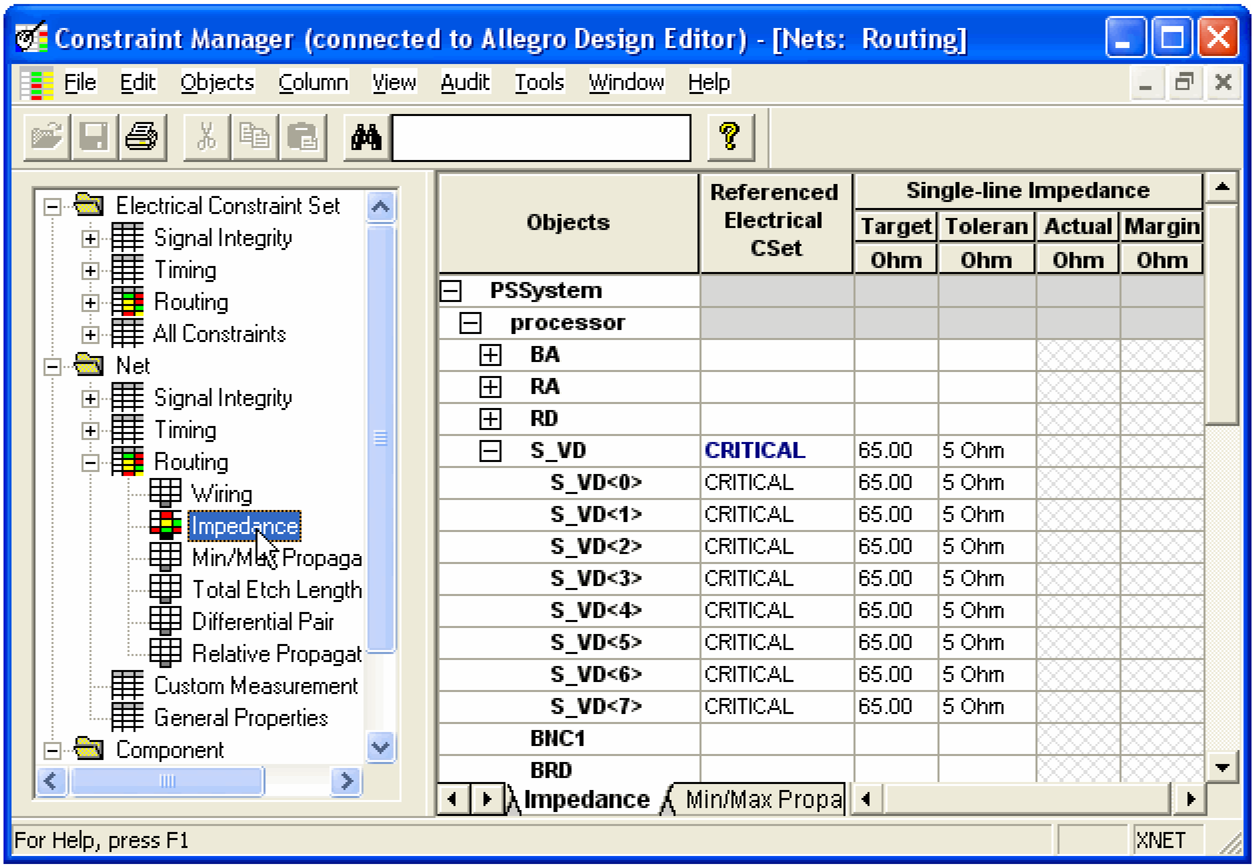
The constraints on the ECSet appear as inherited values on the bits of the bus S_VD.



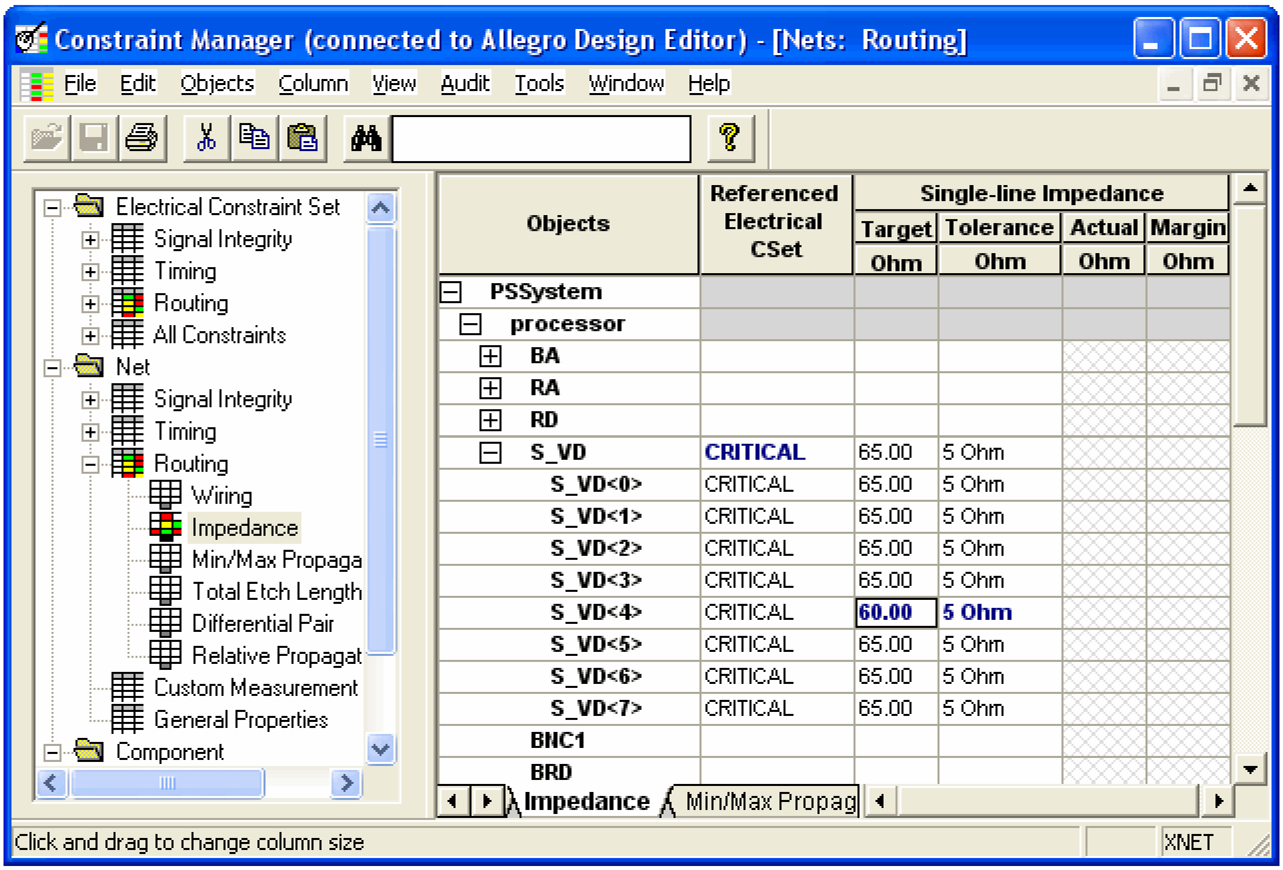
Objects	Referenced Electrical CSet	Topology				Stub Length			Via Count		
		Verify Schedule	Schedule	Actual	Margin	Max mil	Actual mil	Margin mil	Max	Actual	M
PSSystem											
processor											
BA											
RA											
RD											
S_VD	CRITICAL	Yes	Source-load Daisy-chain			150.0			4		
S_VD<0	CRITICAL	Yes	Source-load Daisy-chain			150.0			4		
S_VD<1	CRITICAL	Yes	Source-load Daisy-chain			150.0			4		
S_VD<2	CRITICAL	Yes	Source-load Daisy-chain			150.0			4		
S_VD<3	CRITICAL	Yes	Source-load Daisy-chain			150.0			4		
S_VD<4	CRITICAL	Yes	Source-load Daisy-chain			150.0			4		
S_VD<5	CRITICAL	Yes	Source-load Daisy-chain			150.0			4		
S_VD<6	CRITICAL	Yes	Source-load Daisy-chain			150.0			4		
S_VD<7	CRITICAL	Yes	Source-load Daisy-chain			150.0			4		
BNC1											
BRD											

This is because, constraints assigned on a bus are automatically inherited by the bits of the bus.

18. Click the *Impedance* worksheet in the *Routing* workbook in the *Net* folder.



19. Click in the *Target* field next to the bit `S_VD<4>` and enter `60`.



The constraint values on the bit `S_VD<4>` appear in blue color because the constraints are overridden on the bit.

Summary

You now know how to use create an ECSet and assign it on a net. You also learned the following:

- Constraints assigned on an ECSet are automatically inherited by the objects on which the ECSet is assigned.
- Constraints assigned on a bus are automatically inherited by the bits of the bus.
- How to override an inherited constraint.

For More Information

See:

<u>Allegro Constraint Manager User Guide</u> for more information on working with ECSets.

Lesson 5-3: Assigning Signal Integrity Models

Overview

Design Editor lets you assign signal integrity (SI) models to components and pins in your design during the design capture phase. You can then use SigXplorer to perform topology exploration and analyze the nets in your design for signal integrity issues. This helps you correct signal integrity issues early in the design cycle.

You can manually assign existing signal models to components (such as IC devices) and pins. You can also automatically generate and assign signal models for all two-pin discrete components (resistors, capacitors, and inductors) in your design.

In this lesson, you will learn to assign a signal integrity model on a component and automatically generate models for the two pin discrete components (resistors, capacitors and inductors) in the design.

Multimedia Demonstration

Click the link below to view a Flash-based multimedia demonstration of this lesson.

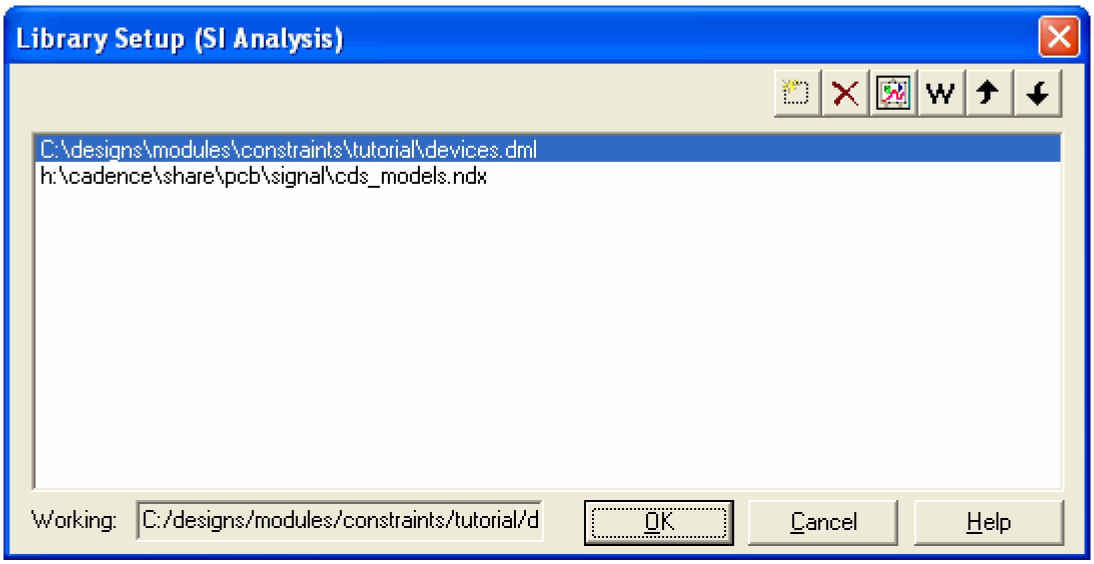
<u>Assigning Signal Integrity Models in Allegro Design Editor</u>

Procedure

Before you can assign signal integrity models on components and pins, you must setup the signal integrity model libraries for your project.

1. In Design Editor, choose *Tools – Signal Integrity – SI Library Setup*.

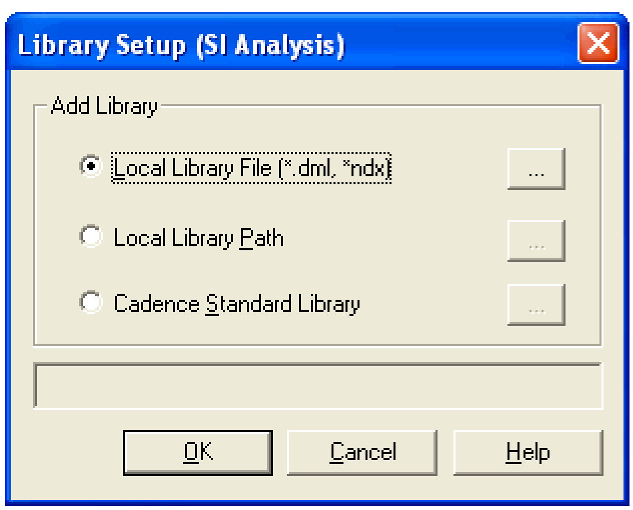
   The Library Setup (SI Analysis) dialog box appears.



   We will now setup a signal integrity model library named `tutorial.dml` for the project.

2. Click .

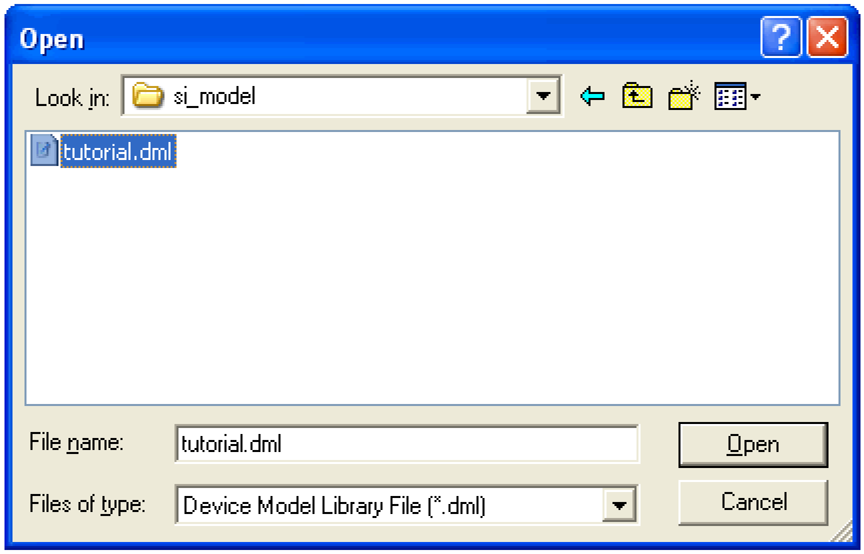
   The Library Setup (SI Analysis) dialog box appears.



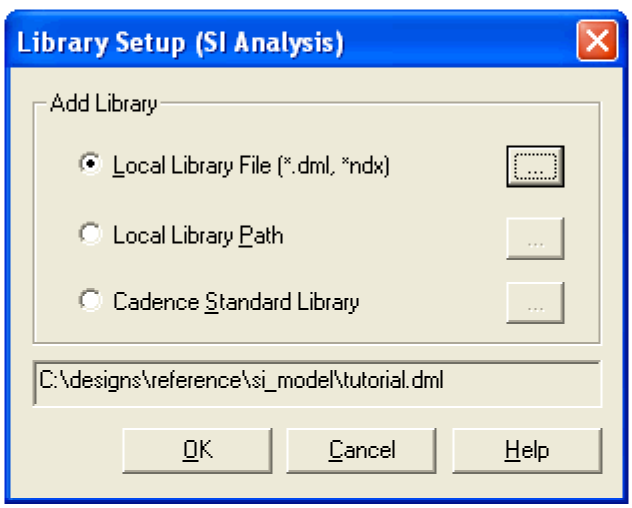
3. Click the browse button next to the *Local Library File [*dml,*ndx]* option.

4. Select the `tutorial.dml` file located at:

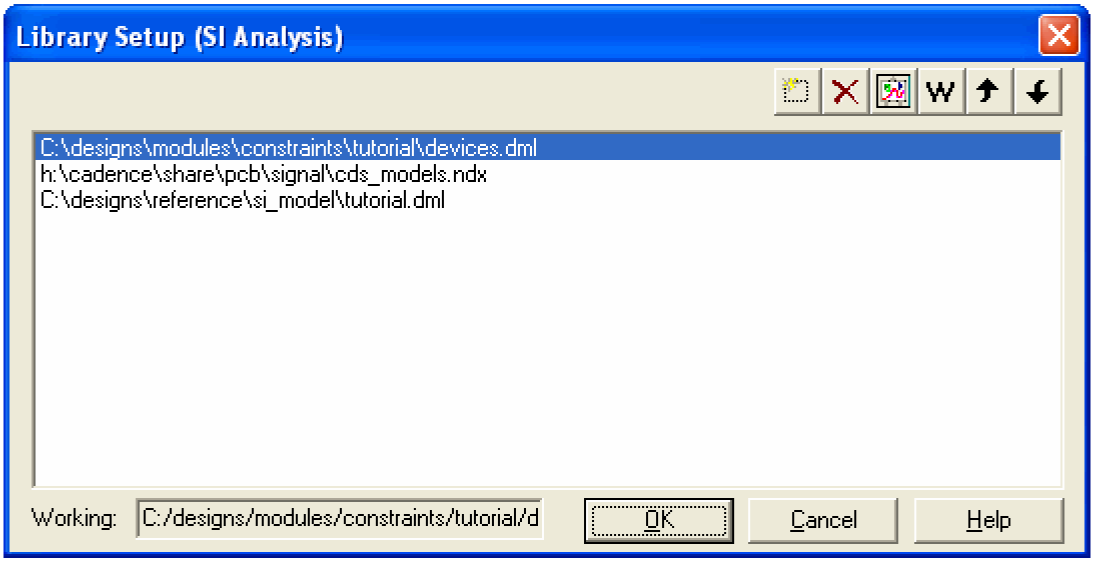
   `<your_work_area>/reference/si_model`



5. Click *Open*.



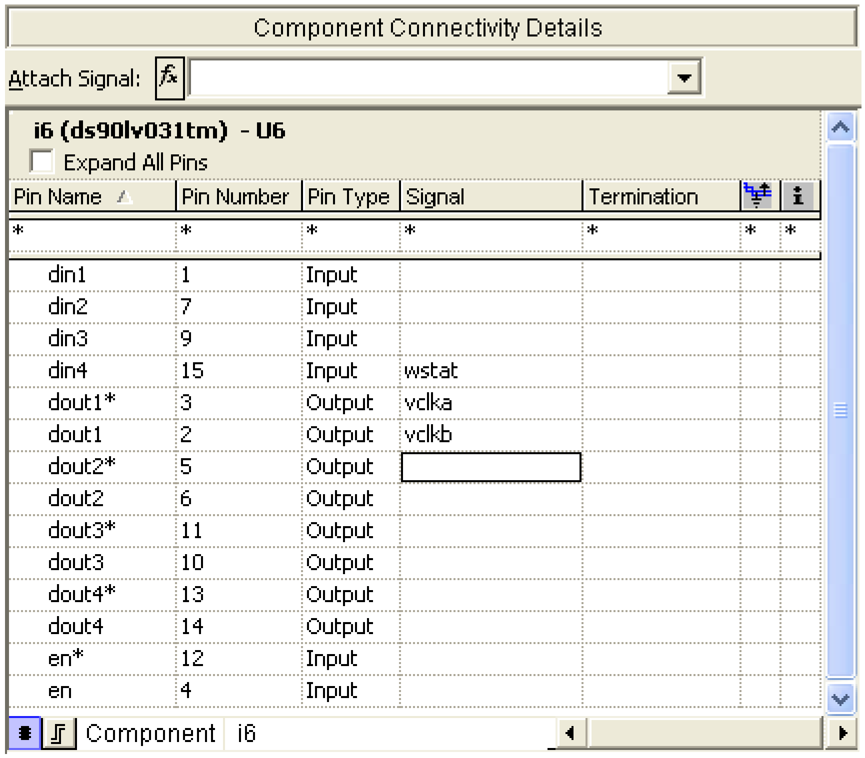
6. Click *OK*.



The `tutorial.dml` model library is displayed in the Library Setup (SI Analysis) dialog box.

7. Click *OK* to close the dialog box.

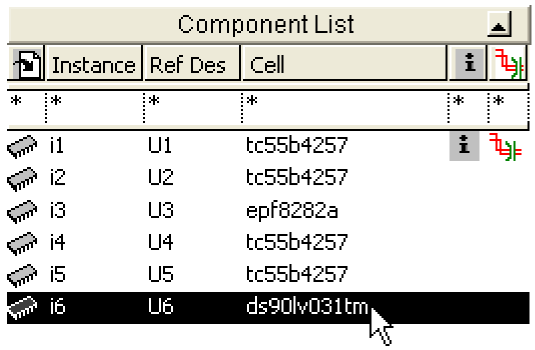
8. Add one instance of the `ds90lv031tm` component from the `classlib` library and connect the pins of the component as shown below.



For more information on adding components in the design and connecting pins to signals, see Module 2: Working with Components and Connectivity.

You will now assign a signal integrity model on the `ds90lv031tm` component and see how Constraint Manager creates a model-defined diff pair object when you assign the signal integrity model.

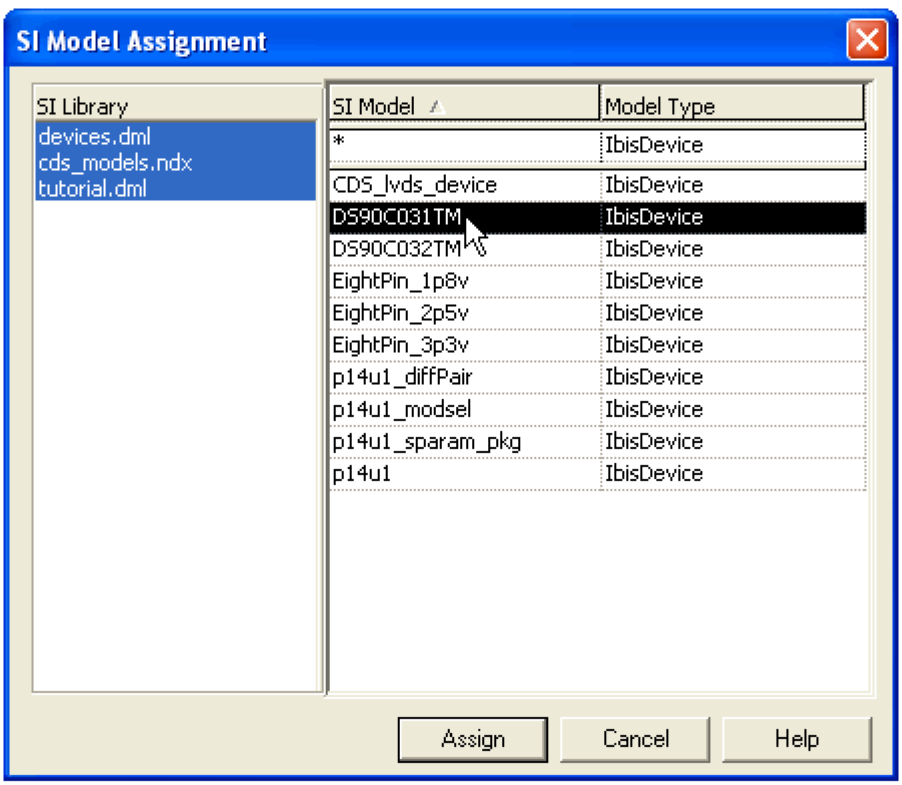
9. Select the `ds90lv031tm` component in the Component List.



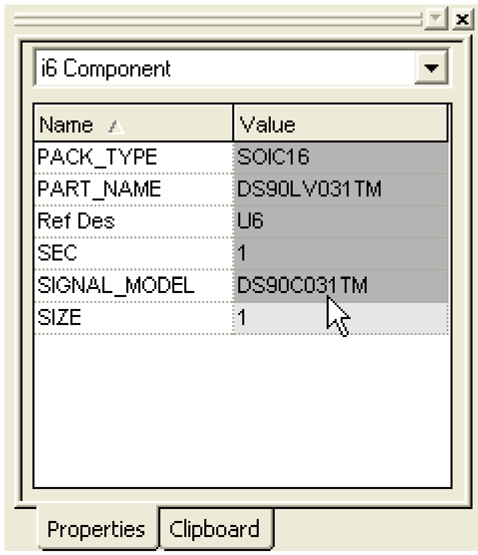
10. Choose *Design – SI Models – Assign Model*.

    The *SI Model Assignment* dialog box appears.

11. Select the IBIS Device model named `DS90C031TM` and click *Assign*.



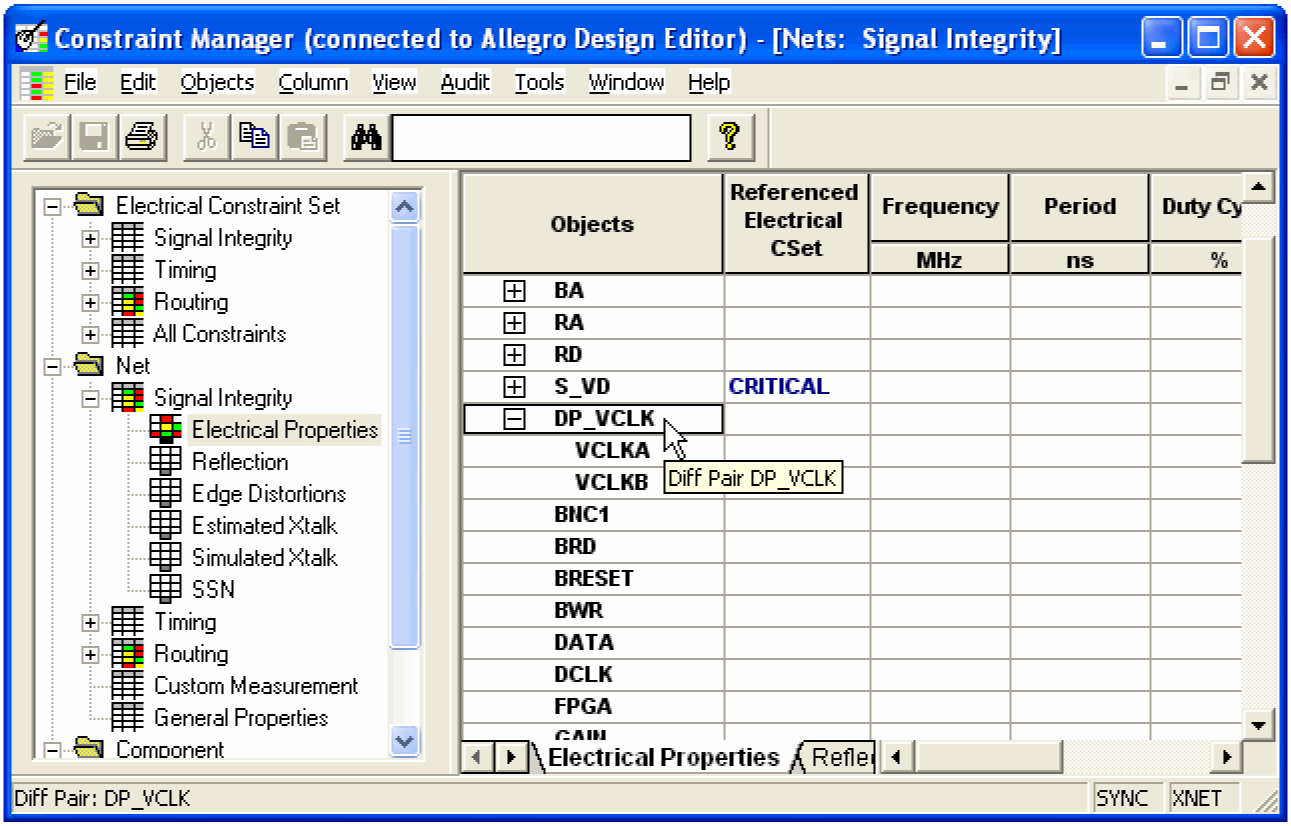
The Properties window displays the `SIGNAL_MODEL` property with the value `DS90C031TM`.



This indicates that the signal integrity model `DS90C031TM` has been assigned on the `ds90lv031tm` component.

12. Click the  toolbar button in Design Editor to open Constraint Manager.

13. Click the *Electrical Properties* worksheet in the *Signal Integrity* workbook in the *Net* folder.



Note that a model-defined diff pair named DP_VCLK with the member nets VCLKA and VCLKB has been automatically created in Constraint Manager.

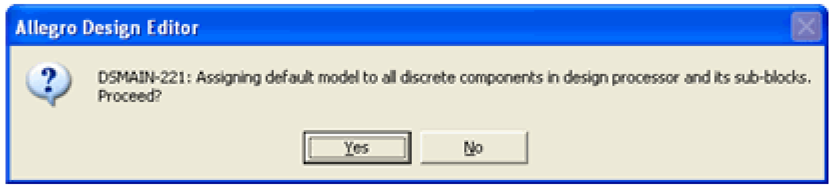
Design Editor allows you to automatically generate and assign signal integrity models for all two-pin discrete components (resistors, capacitors, and inductors) in your design.

When you apply terminations in the design, signal integrity models are automatically assigned to the resistors and capacitors used in the termination. You can automatically generate and assign signal integrity models for all other two-pin discrete components such as resistors, capacitors, and inductors used in the design.

You will now automatically generate and assign signal integrity models to all two-pin discrete components in the design.

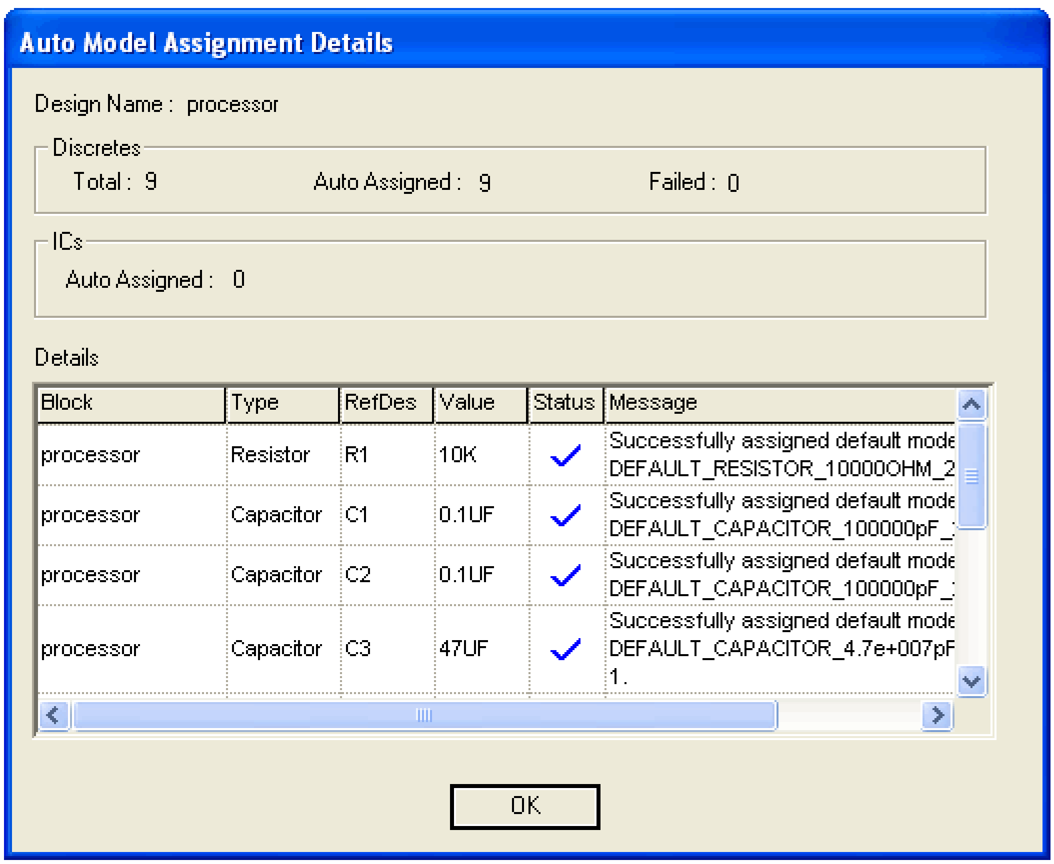
14. Choose *Tools – Signal Integrity – Auto Assign Discrete Models*.

The following message box appears.



15. Click *Yes*.

The Auto Model Assignment Details dialog box appears displaying the details of the model assignments made on two-pin discrete components in the design.



16. Click *OK* to close the dialog box.

Summary

You now know how to assign signal integrity models on components in Design Editor. You also learned how to automatically assign signal integrity models on two pin discrete components used in the design.

For More Information

See:

Working with Signal Integrity Models chapter of *Allegro Design Editor User Guide*.

Lesson 5-4: Applying Constraints from SigXplorer

Overview

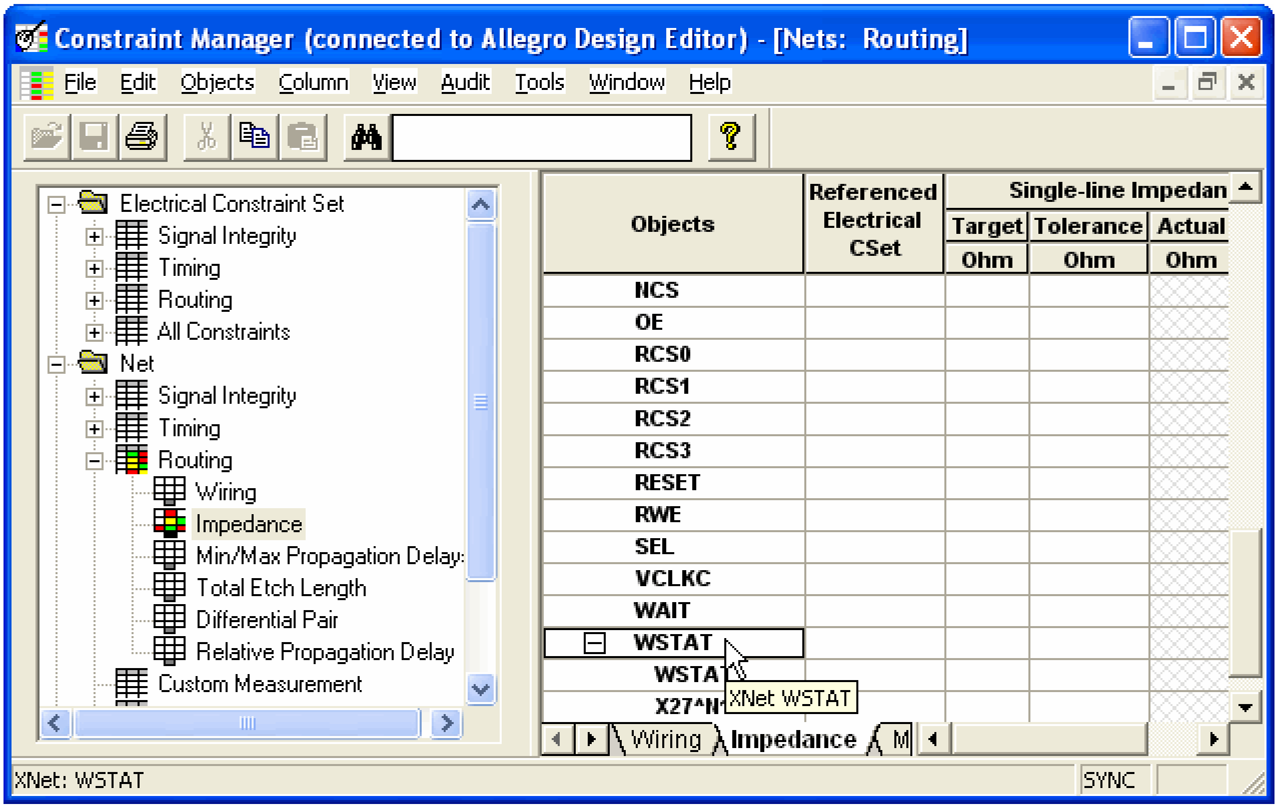
You can use SigXplorer to analyze the high speed nets in your design for signal integrity issues and create a set of constraints for the nets. The topology file containing these constraints becomes an ECset. You can then apply the ECSet (topology file containing constraints) to nets in the design.

In this lesson you will learn to extract a net into SigXplorer from Constraint Manager, set constraints in SigXplorer and then apply the topology containing the constraints on the net in Constraint Manager.

Procedure

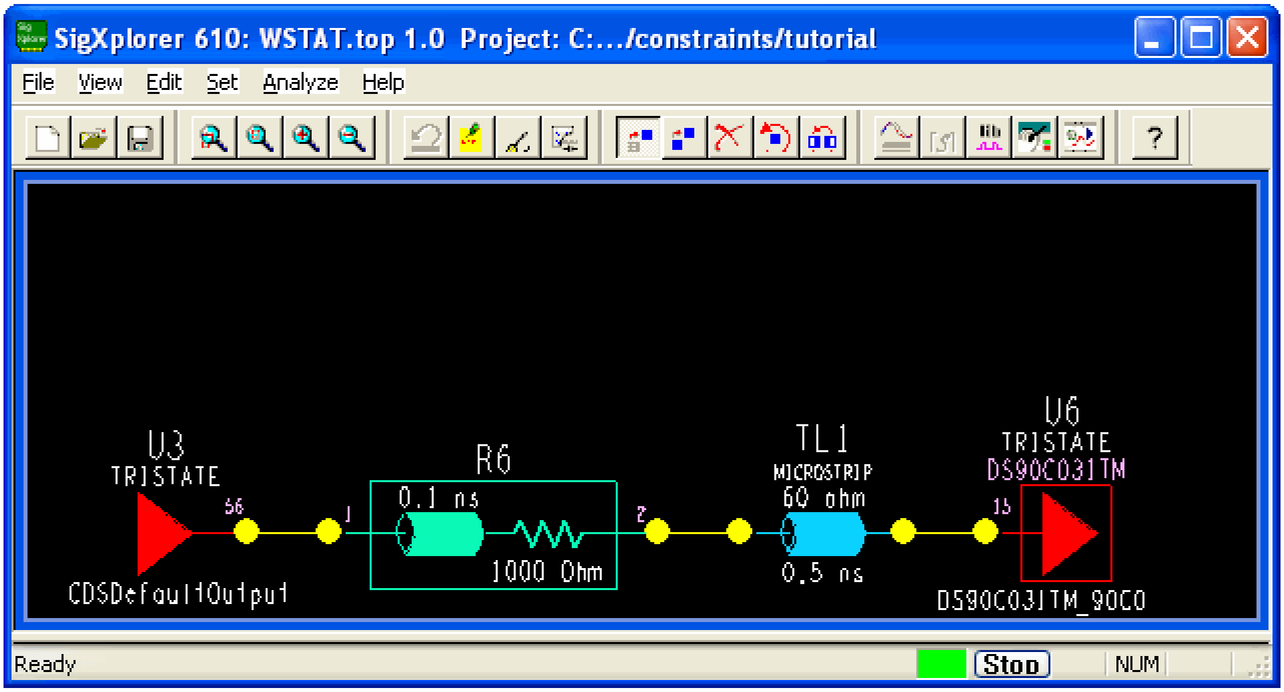
1. Click the  toolbar button in Design Editor to open Constraint Manager.
2. Click the *Impedance* worksheet in the *Routing* workbook in the *Net* folder.

3. Select the Xnet named WSTAT.



4. Choose *Tools – SigXplorer*.

The topology for the net is displayed in SigXplorer.

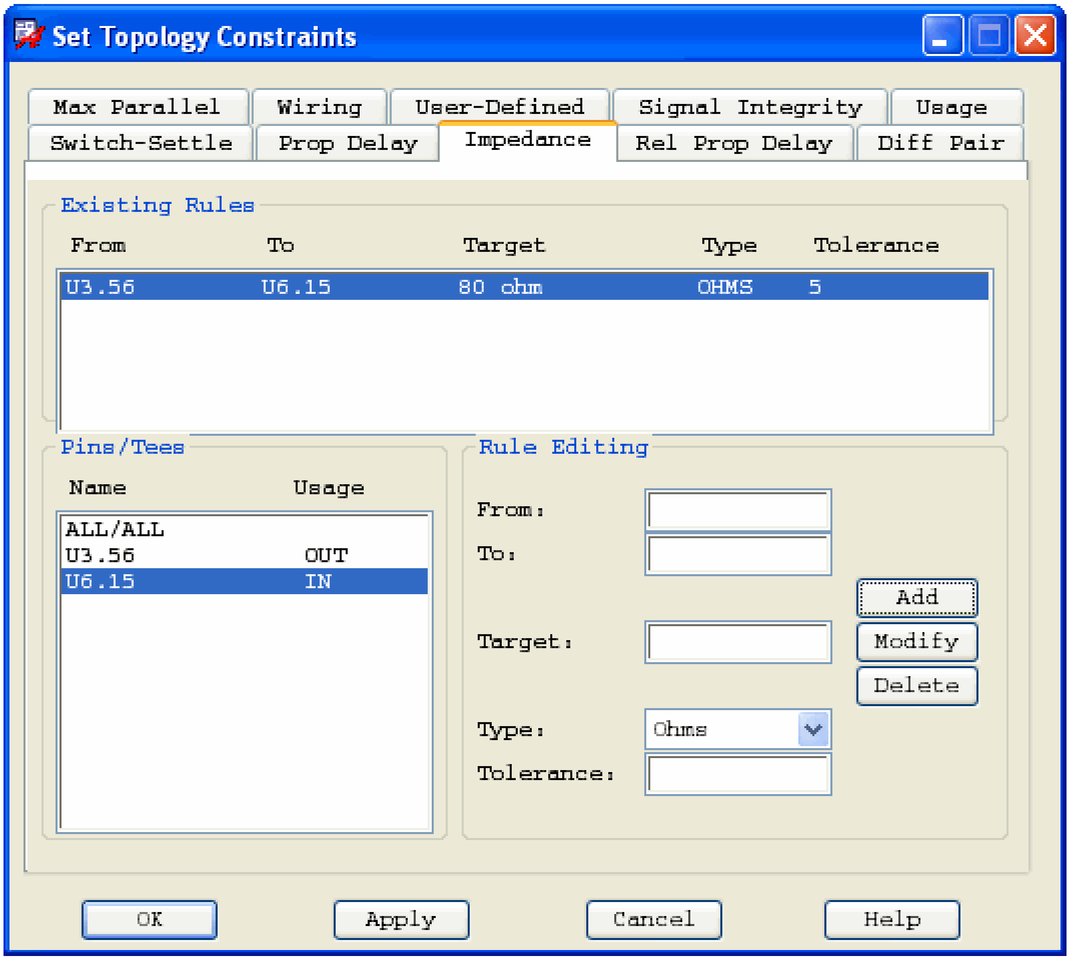


5. Choose *Set – Constraints*.

   The Set Topology constraints dialog box appears.

6. Click the *Impedance* tab.
7. In the *Pin/Tees* list, click on `U3.56`.
8. In the *Pin/Tees* list, click on `U6.15`.
9. In the *Target* field, enter `80`.
10. In the *Tolerance* field, enter `5`.

11. Click *Add*.

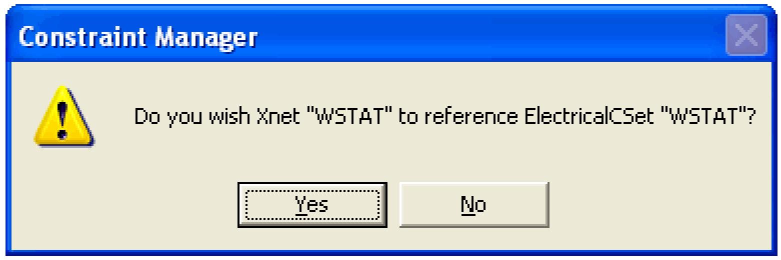


12. Click *OK* to save the changes and close the dialog box.

13. Choose *File – Update Constraint Manager*.

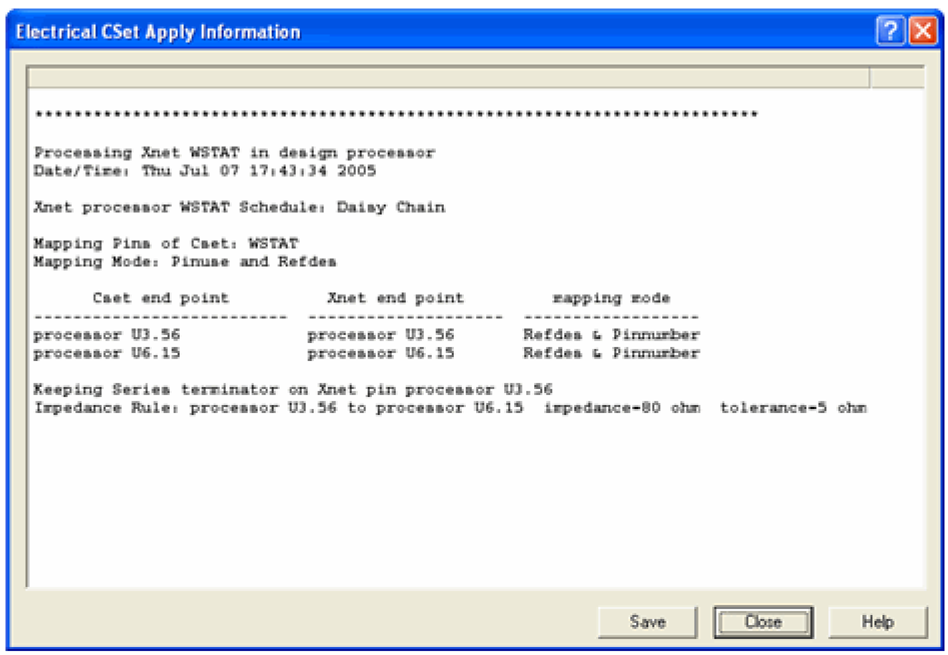
An ECSet named `WSTAT` with the constraints you added in SigXplorer is created.

The following message box appears.

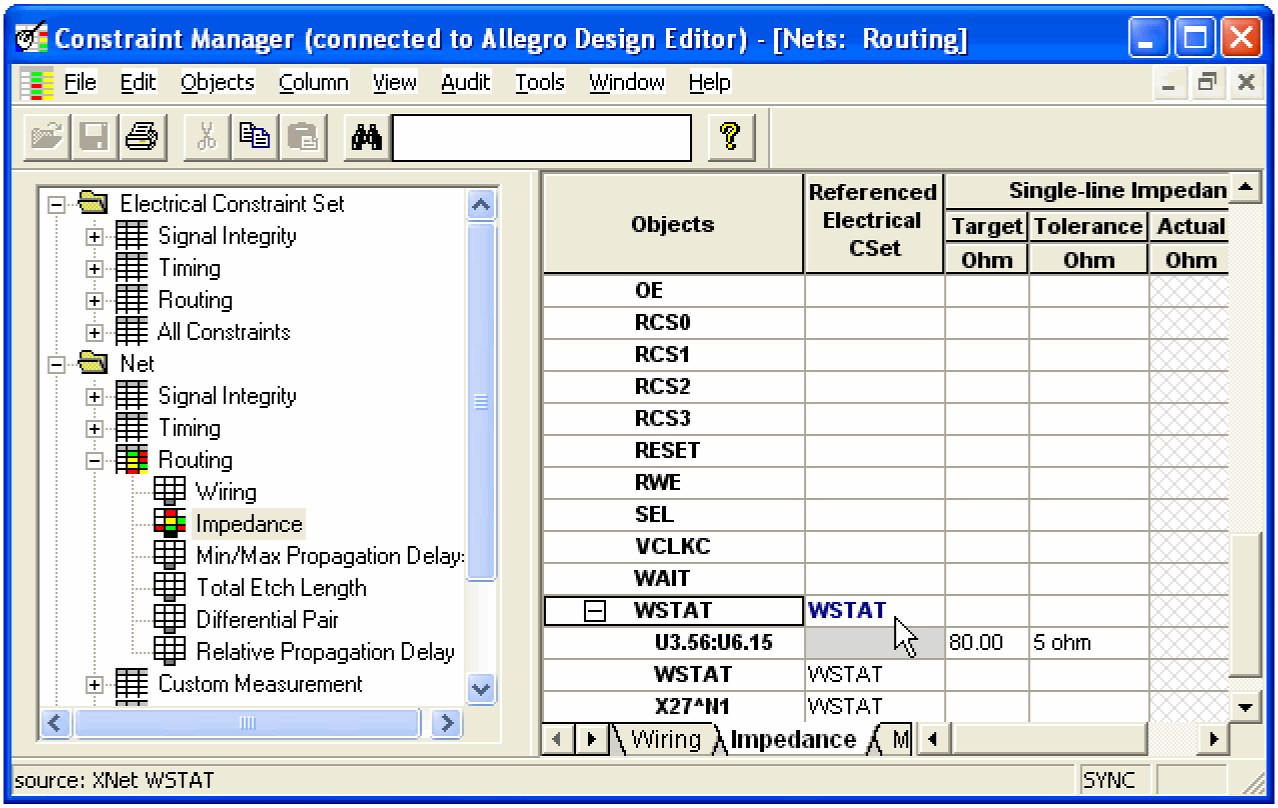


14. Click *Yes*.

The Electrical CSet Apply Information dialog box appears.



15. Click *Close*.



In the *Impedance* worksheet note that the ECSet `WSTAT` has been applied on the XNet `WSTAT`.

You can now assign the ECSet `WSTAT` on other XNets in Constraint Manager.

Summary

You now know how to extract a net into SigXplorer from Constraint Manager, set constraints in SigXplorer and then apply the topology containing the constraints on the net in Constraint Manager.

For More Information

See:

- Allegro Constraint Manager User Guide for more information on using SigXplorer with Constraint Manager.
- Allegro PCB SI SigXplorer User Guide for more information on using SigXplorer.

Module 6: Creating a Hierarchical Design

Prerequisite

To work with the lessons in this module, open the `hier_design.cpm` project located at `<your_work_area>\modules\hier_design\hier_design` in Design Editor.

For more information, see Understanding the Sample Design Files on page 14.

Lessons

This module consists of the following lessons:

Multimedia Demonstration

Click the link below to view a Flash-based multimedia demonstration of this module.

Working with Hierarchical Designs

Completion Time

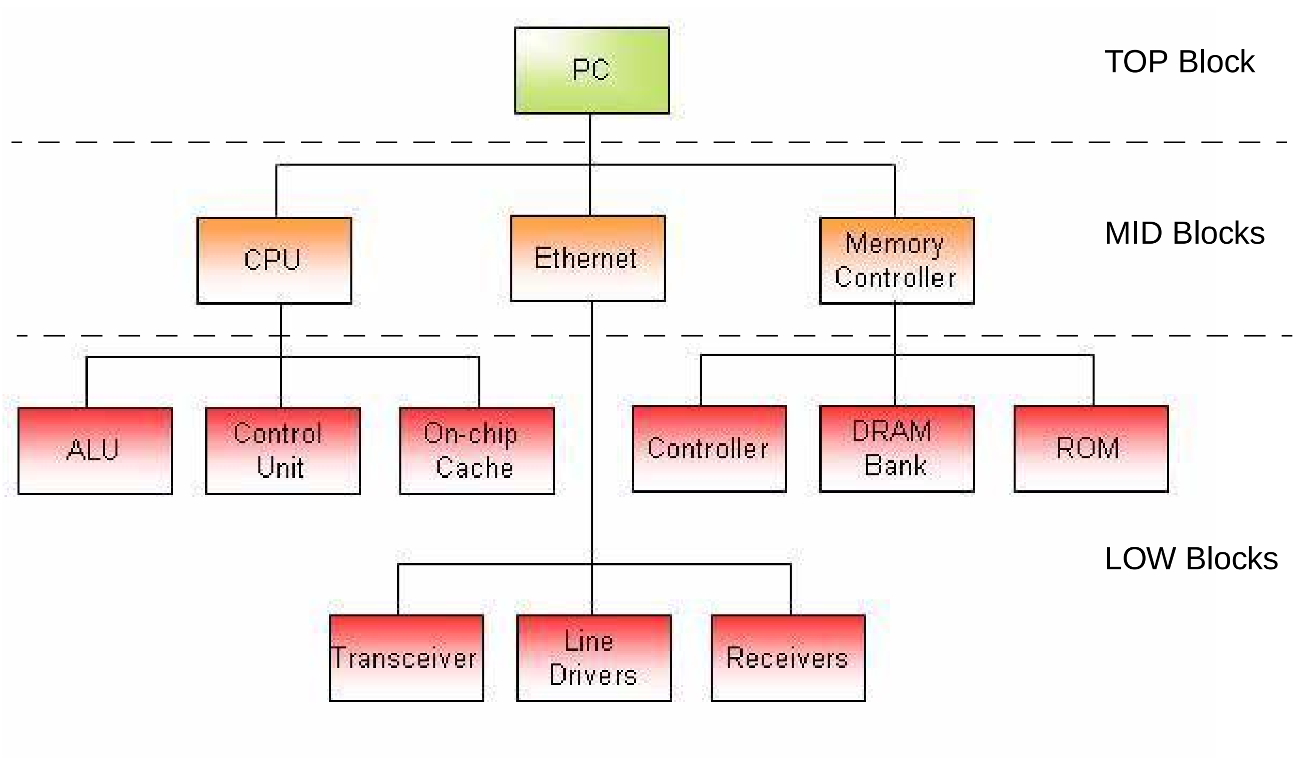
- 2 hours for written lessons
- 50 minutes for multimedia demonstrations

Overview

You can use a hierarchical design structure to divide a design into sub designs or blocks, where each block represents a logical function.

An example of a hierarchical design is shown below.

Figure 6-1 Hierarchical Design Example



Notice that the design has a top-level block named `PC`. This block includes three sub-blocks, `CPU`, `Ethernet`, and `Memory Controller`. Each of these sub-blocks includes more lower-level blocks. In the hierarchical design example, the `CPU` block is divided into three sub-blocks named `ALU`, `Control Unit` and `On-chip Cache`.

To create a hierarchical design, you can use the top-down or bottom-up design methodology.

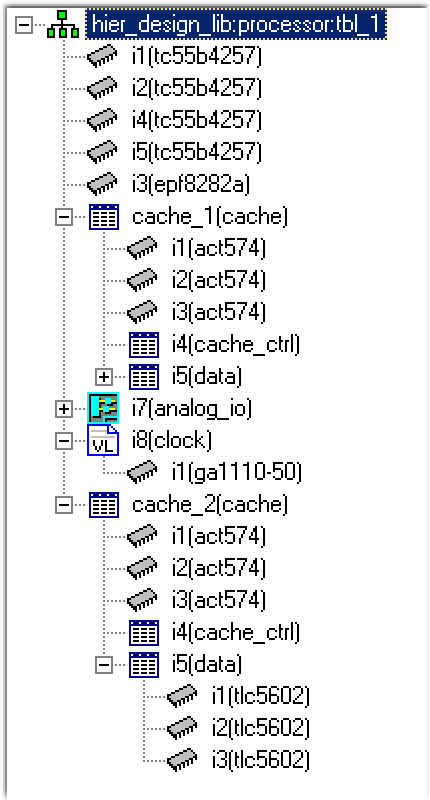
- Top-Down Methodology—In the top-down methodology, you first create the top-level design PC. Next, based on the logical partitioning of the design you create blocks for each logic function. In the case of the PC design, the top-level design will have three blocks: CPU, Ethernet and Memory Controller. After creating the top-level design with the necessary blocks, you can create the lower-level blocks.
- Bottom-Up Methodology—In the bottom-up methodology, you create a lower-level block first. For the design `PC`, you can first create the designs for the lowest level of hierarchy—`ROM`, `DRAM Bank`, and `Memory Controller`. You can then create the higher level design `Memory Controller`. You can then repeat the process to create `Ethernet` and `CPU` blocks, and then create top-level design by integrating the `Memory Controller`, `Ethernet` and `CPU` blocks into the `PC` design.

While creating a hierarchical design, you can create blocks by:

- Creating them as stand-alone blocks that are part of a library.
- Instantiating them directly into the design. These blocks are integrated as sub-circuits in the hierarchical design.

Allegro Design Editor enables you to seamlessly create hierarchical designs using top-down and bottom-up design methodologies. You can have a combination of spreadsheet, Verilog and schematic blocks in your design. The use of multiple design blocks will allow you flexibility in capturing an electronic circuit. For example, you may prefer to use spreadsheet blockS to capture large pin-count devices and schematic blocks to capture analog designs.

In this module, you will create the following hierarchical block:



You will start by creating design blocks. In the first three lessons of this module, you will create spreadsheet (`cache`), schematic (`analog_io`) and Verilog (`clock`) blocks. In the fourth and fifth lessons, you will learn to add components and set packaging options such that components in different blocks get unique reference designators. You will also learn the differences in editing properties or connectivity in the Context or Master modes. Finally, you will learn to create hierarchical designs by adding blocks within other blocks using both top-down and bottom-up methodologies.

Lesson 6-1: Creating a Spreadsheet Block

Overview

In this lesson, you will learn to create a spreadsheet block and instantiate it in an existing design.

You will create a spreadsheet block named `cache` and place it in the `hier_design_lib` library. This logical block (in spreadsheet format) will have the following input/output ports:

- Input ports—`vd<7..0>`, `gain`, `vclka`, and `vclkb`
- Output ports—`outa` and `outb`

Procedure

1. In the `hier_design.cpm` project, notice that the top-level design name or root design name is `processor`. The Hierarchical Viewer displays the root design.

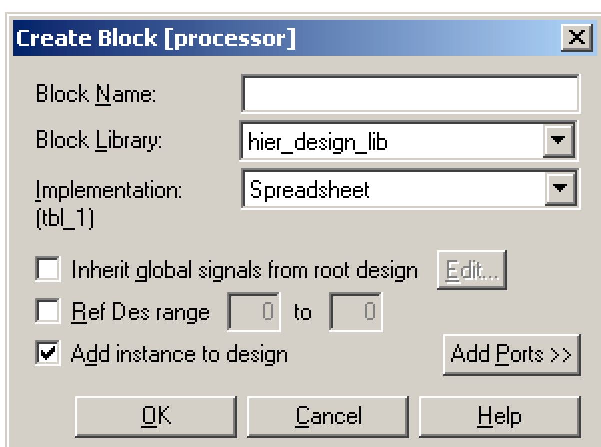
   The Hierarchy Viewer provides you a tree view of the complete design hierarchy and lets you quickly access all the blocks and components in your design.

   You can use the Hierarchy Viewer to view the binding of any block.

   You will now create a new spreadsheet block named `cache.`

2. Choose *Design - Create Block*.

   The Create Block dialog box appears. You use this dialog box to create blocks—Spreadsheet, Verilog or Schematic—in Allegro Design Editor.

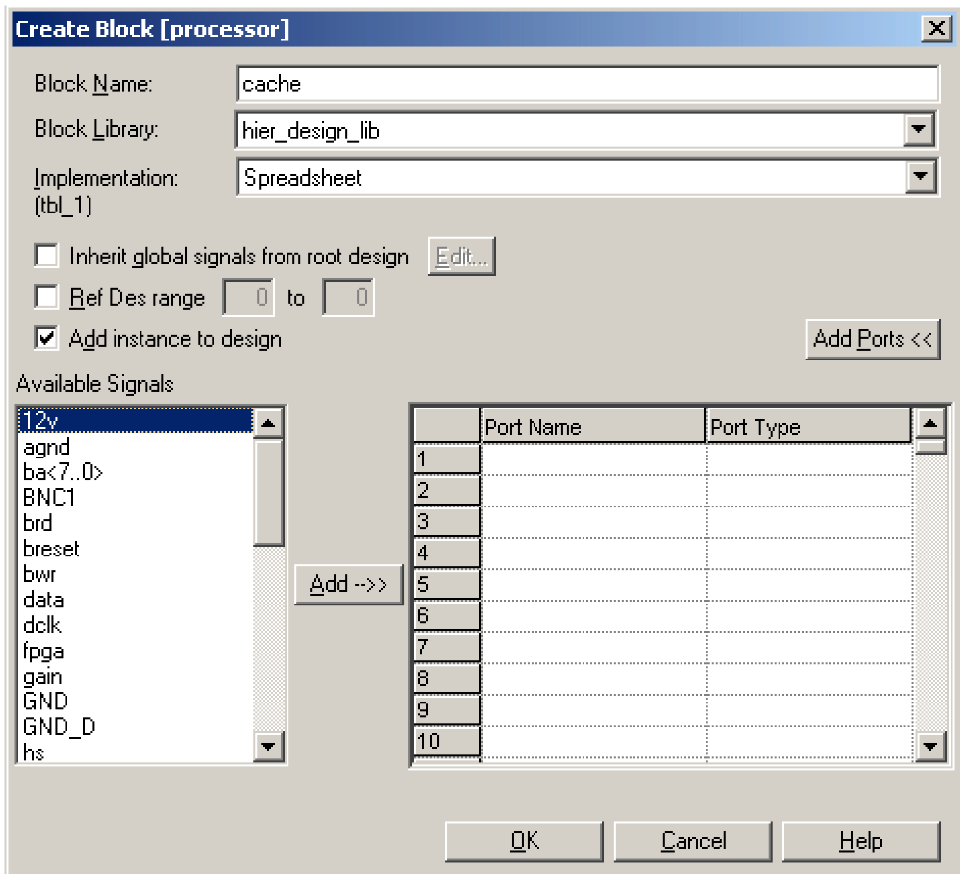


3. Type `cache` in the *Block Name* field.

The *Block Library* field shows that the new block being created will be added in the `hier_design_lib` library. If required, you can change the library name.

4. Click the *Add Ports* button to display the port list.

   The Create Block dialog box expands to display the port list.



You can assign port names and define the port type as `IN`, `OUT`, or `INOUT`.

5. To add an input port `vd<7..0>`, type `vd<7..0>` in the *Port Name* field and press `Tab` to move to the *Port Type* field.

6. Type `IN` or select it from the *Port Type* drop-down list.

7. Repeat steps step 5 and step 6 to add the following input ports—`gain`, `vclka`, and `vclkb`.

8. To add an output port `outa`, type `outa` in the *Port Name* field and choose `OUT` from the *Port Type* drop-down list.

9. Repeat step 8 to add another output port, `outb`.

   You can inherit global signals available in the parent (`processor`) block in the current block.

10. Select the *Inherit global signals from root design* check box.

11. Click the *Edit* button.

    The Global Signals dialog box appears displaying the list of global signals that you can inherit from the `processor` block.

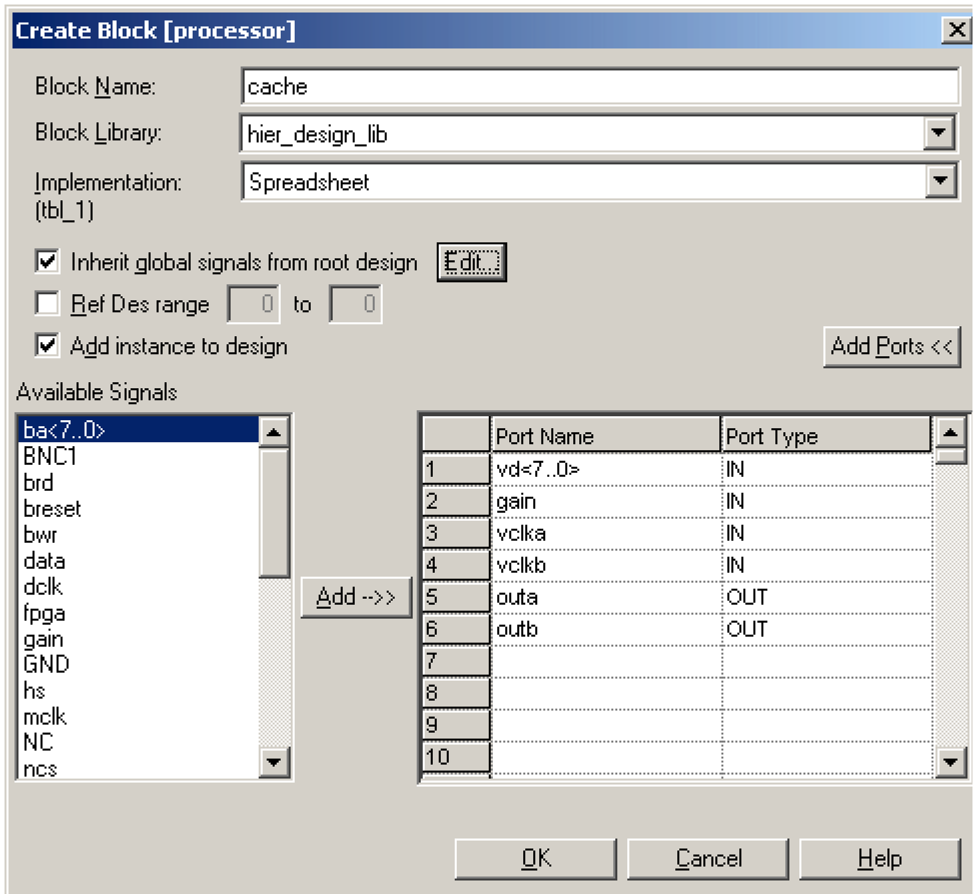
12. Without making any changes to the list, click *OK.*

13. Ensure that the *Add instance to design* check box is selected. This will add an instance of the new block in the `processor` design.

Tip

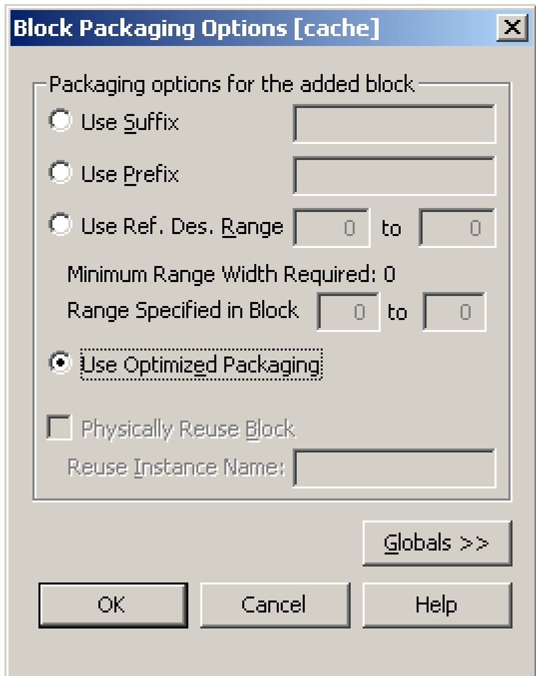
You can also add instances of blocks using the Component Browser.

At this point, the Create Block dialog box should display the following settings:



14. Click *OK* to store the `cache` block in the `hier_design_lib` library.

The `cache` block is stored in the `hier_design_lib` library. The Block Packaging Options dialog box appears.

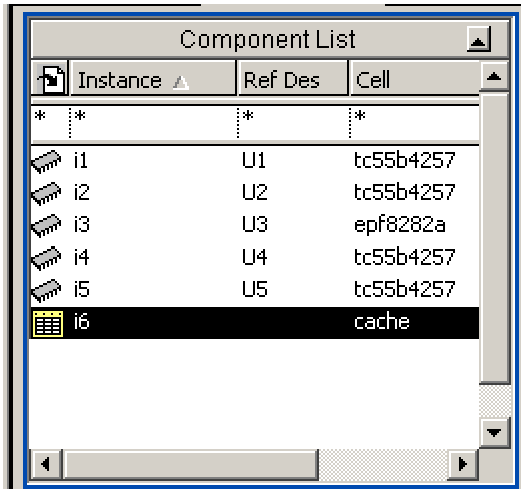


15. Ensure that the *Use Optimized Packaging* option button is selected. This option will use optimized reference designators in context of the root block.

    Note: The Block Packaging Options dialog box provides you a powerful mechanism of renaming reference designators and physical net names while integrating a block into any design. You can also alias a global signal in the block to a signal in the design in which you are adding the block. You will learn to use different block packaging options across this module.

16. Click *OK*.

A new block named `cache` is added in the Component List and the Hierarchy Viewer. The block is assigned the instance name `i6`. If required, you can change the instance name.



17. Double-click the `i6` instance of the `cache` block in the Hierarchy Viewer to open it.

    The `cache` block appears. All signals in the `cache` block are displayed in the Signal List. This block does not contain any component. You will add components to the `cache` block and identify the advantages of different packaging options in Lesson 6-4: Setting Up Block Packaging Options on page 233.

18. Close the `cache` block by selecting *File - Close*.

Summary

In this lesson, you learned to create a spreadsheet block in an existing design.

For More Information

See:

Working with Hierarchical Designs chapter of *Allegro Design Editor User Guide*.

Lesson 6-2: Adding a Schematic Block in a Design

Overview

In this lesson, you will learn to add a schematic block in a spreadsheet design.

You will create a schematic block `analog_io` from Spreadsheet Editor by importing components from an already existing schematic design.

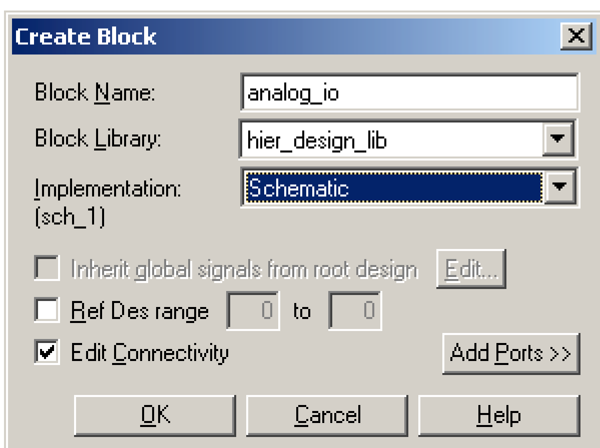
Procedure

1. Choose *Project - Create Block.*

   The Create Block dialog box appears.

2. Specify the block name as `analog_io`.
3. From the *Implementation* drop-down list, choose *Schematic*.
4. Select the *Edit Connectivity* check box.

   The Create Block dialog box should look like:



5. Click *OK*.

   The `analog_io` block is opened in Design Entry HDL.

   You can create a new schematic here. Since creating a schematic in Design Entry HDL is beyond the scope of this tutorial, you will create the `analog_io` schematic block by copying parts from an existing schematic.

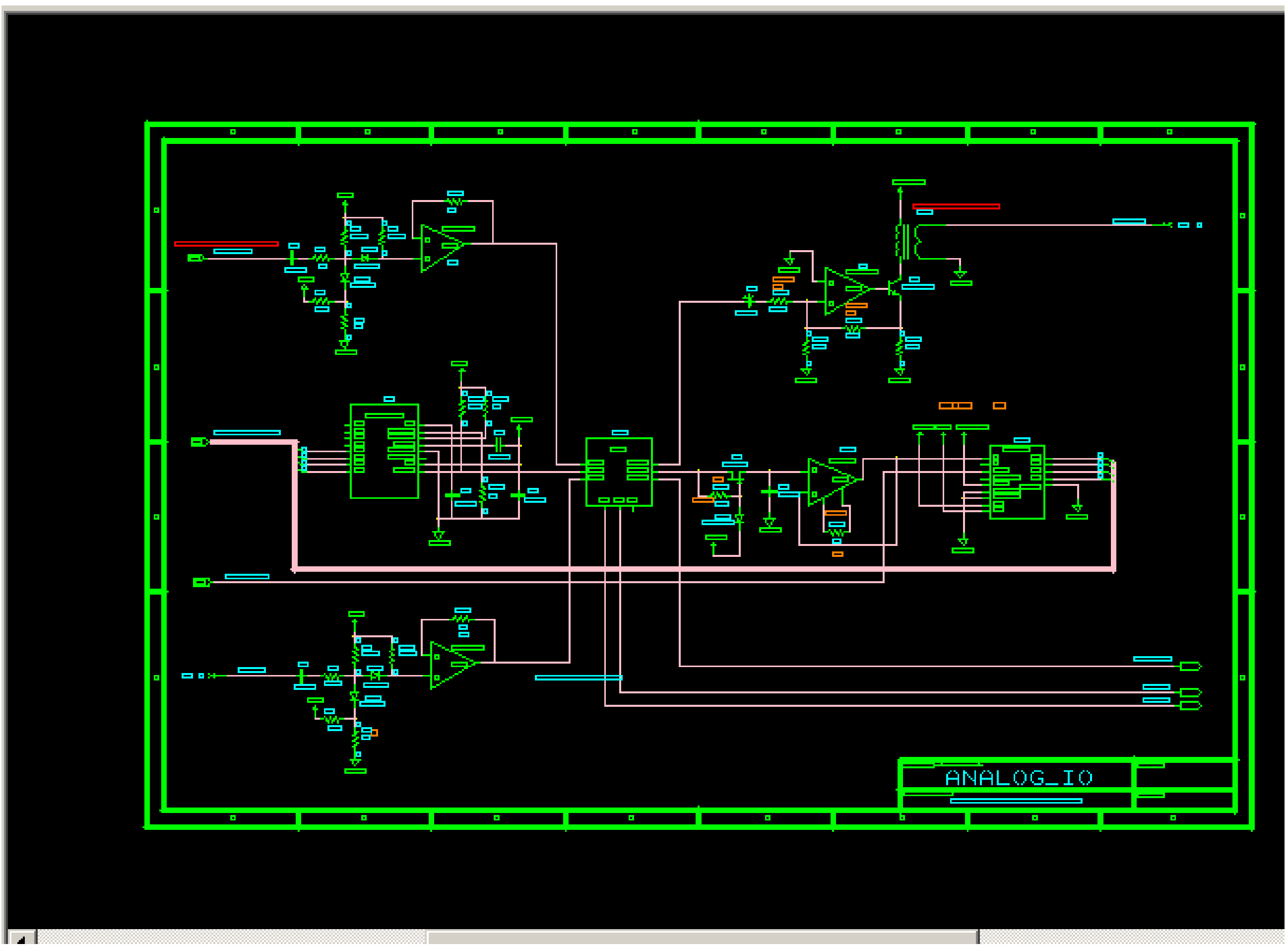
6. Open the `analog_io.cpm` located at *`<your_work_area>/`*`reference/schematic/analog_io` in Allegro Design Entry HDL.

7. Select *Group - Create - By Rectangle*.

   The Design Entry HDL Console window displays the message:

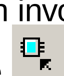
   ```
   select A
   Using group "A"
   ```

8. Click the top-left page border and drag the mouse to the bottom-right page border and click again.



All components and their connectivity, including the page border are selected. The Console window displays the message:

```
Group "A" contains:
   82 bodies   730 properties   0 notes   163 wires   25
dots   0 images
```

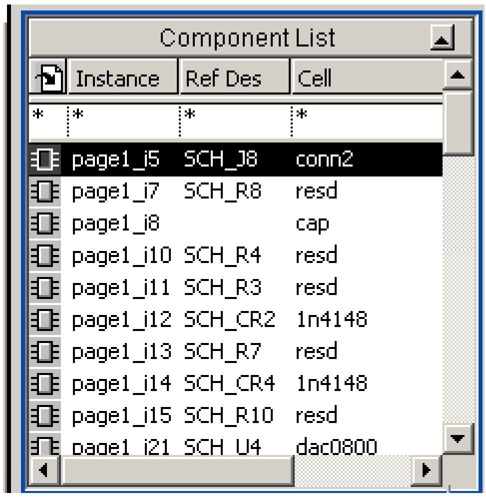
9. Select *Group - Copy All [A]*.
10. Select the other instance of Design Entry HDL that has the `analog_io` schematic.
11. Select *Edit - Paste* to paste the contents of the group A created in the other schematic into this schematic.
12. Select *File - Save All* to save the schematic.
13. Select *File - Exit* for both invocations of the schematic.
14. In Design Editor, click the  *Add Component* tool button.
15. Select the `analog_io` cell in the `hier_design_lib` library.
16. Click the *Add* button and close the Component Browser.

    The Block Packaging Options dialog box appears.
17. Select the *Use Prefix* option button, enter `SCH` in the *Prefix* field and click *OK*.

    A new block `analog_io` is added in the Hierarchy Viewer and it also appears in the Component List. Notice that the icon placed next to the block represents block type as schematic as shown below.

18. Select *File - Save All* to save the design.
19. Double-click the `analog_io` block in the Hierarchy Viewer to open it in the Context mode.

    The `analog_io` appears within the Spreadsheet Editor. Notice that all components are listed in the Component List and all signals are listed in the Signal List. Also notice that the reference designator of each component is prefixed with `SCH`.



Note: The schematic block opens in a read-only view in the Spreadsheet Editor. For more information about how to use read-only blocks of type schematic in Design Editor, see the *Working with Block Designs* chapter of the *Allegro Design Entry HDL User Guide*.

Summary

In this lesson, you learned to add a schematic block in Allegro Design Editor.

For More Information

See:

Working with Hierarchical Designs chapter of *Allegro Design Editor User Guide*.

Lesson 6-3: Adding a Verilog Block

Overview

In this lesson, you will learn to add a Verilog block in Allegro Design Editor.

Concept

You can capture your design in Verilog by creating Verilog blocks. You can also import Verilog files to create Verilog blocks. The modules in a Verilog file are imported as blocks. In this lesson, you will learn to create a Verilog block named `clock` and add components and connectivity in it.

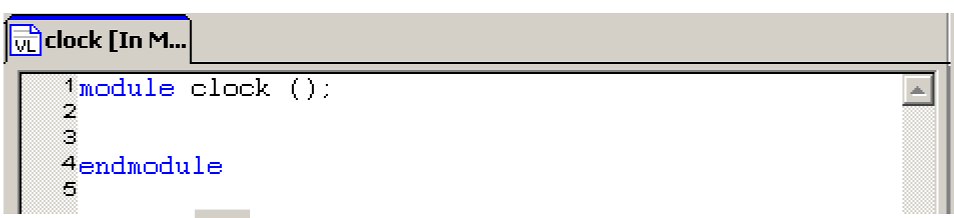
Procedure

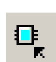
1. Choose *Project - Create Block.*

   The Create Block dialog box appears.

2. Specify the block name as `clock`.
3. From the *Implementation* drop-down list, choose *Verilog*.
4. Select the *Edit Connectivity* check box.
5. Click *OK*.

   A new tab listing the content of the `clock` block in a Text Editor appears. Notice the Verilog defining the module `clock` is displayed.

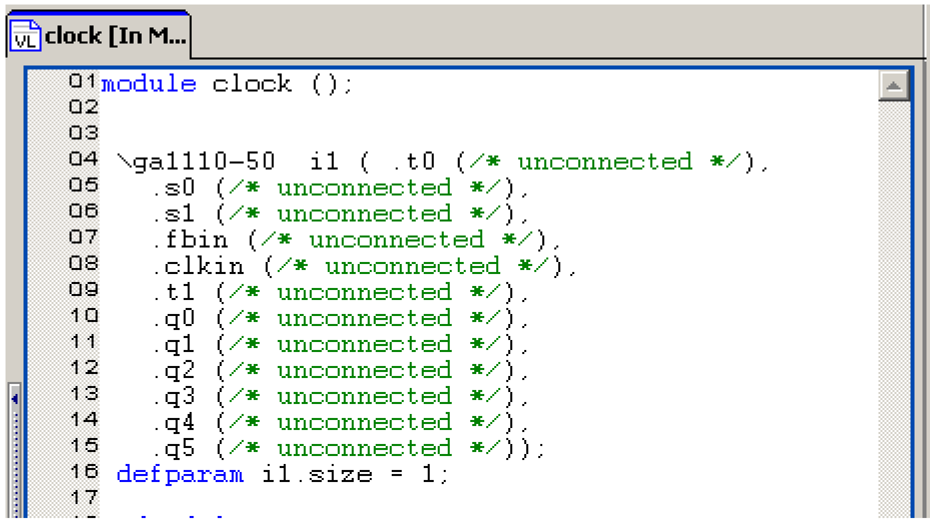


6. Click the  *Add Component* toolbar button.
7. Select the `ga1110-50` component from the `analog` library.

8. Select the PPT row corresponding to the `ga1110-50` component.

9. Click *Add* and close the Component Browser.

   The `ga1110-50` component appears.

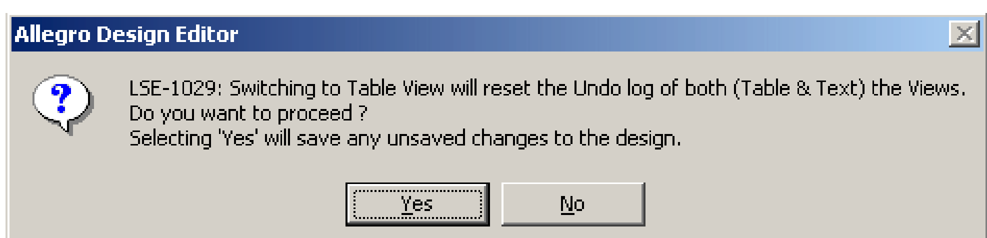


The Verilog file appears in a read-only text viewer. Notice that all pins are unconnected.

You cannot edit the Verilog file in the text viewer. However, you can quickly make pin-signal connections using the Spreadsheet Editor.

10. Select *Design - Edit Connectivity in Table Editor* to use the Spreadsheet editor for editing the Verilog design.

    A message box informing that you may not be able to undo any operation by switching to Table view appears.



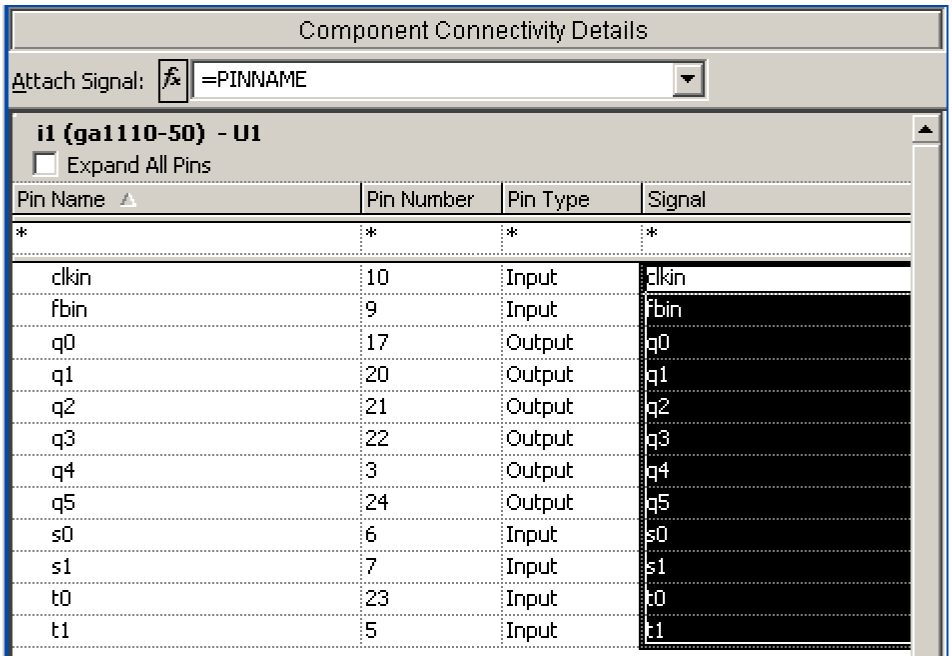
11. Click *Yes*.

    The Verilog design is displayed in the Spreadsheet Editor.

12. In the Component Connectivity Details pane, select the cells in the Signal column next to all the pins.

13. Select the `=PINNAME` function in the *Attach Signal* field to assign the same signal name as the pin name.

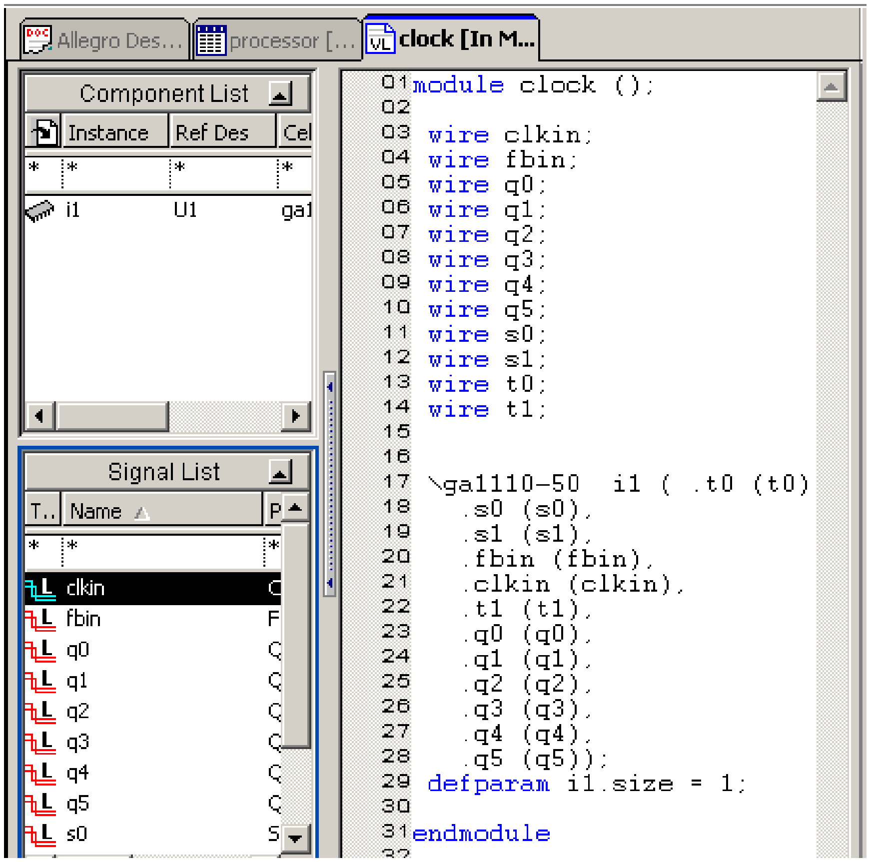
    New signals with the same names as pin names are connected to the pins as shown below:



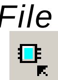
14. Select *Design - Edit Connectivity in Text Editor* to use the Text editor for viewing the Verilog design.

15. Click *Yes* to close the message box.

The Verilog block now has detailed component and connectivity information.



Notice that new signal connections are made for the `ga1110-50` component. The new wire names also appear in the *Signal List.* By switching between table and text views, you can quickly edit connectivity in a Verilog block.

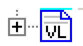
16. Choose *File - Save All* to save the changes in the `clock` block.
17. Choose *File - Close* to close the tab for the `clock` block.
18. Click the  *Add Component* tool button.
19. Select the `clock` cell.
20. Click the *Add* button and close the Component Browser.

    The Block Packaging Options dialog box appears.

21. Select the *Use Suffix* option button and enter V in the *Suffix* field.

22. Click *OK*.

    A new block `clock` is added in the Hierarchy Viewer and it also appears in the Component List. Notice that an icon appears next to the block. It represents block type as Verilog as shown below.

23. Select the `clock` block in the Hierarchy Viewer.
24. Click the  *Descend* tool button.

    A new tab for the `clock` block opens. Notice that the reference designator of all components within this block and the physical net names of all signals have the V letter appended.

    The Verilog code is also displayed in the Text Editor.

25. Choose *File - Save All* to save the changes in all blocks.
26. Choose *File - Close* to close the Verilog block.

Summary

In this lesson, you learned to add a Verilog block in Design Editor.

For More Information

See:

<u>Working with Hierarchical Designs</u> chapter of *Allegro Design Editor User Guide*.

<u>Capturing Verilog Designs</u> chapter of *Allegro Design Editor User Guide*.

Lesson 6-4: Setting Up Block Packaging Options

Overview

In this lesson, you will learn to add components in a hierarchical block. You will also use different packaging options to ensure that unique reference designators are assigned to different components or blocks.

Concept

You can use the Component Browser to add components or blocks to any design.

When you add any block, the Block Packaging Options dialog box appears. In this dialog box, you can define a suffix, prefix, or reference designator range that can uniquely define the reference designators for the components in the block.
The use of these options help you:

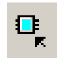
- Avoid packaging errors by ensuring that the same reference designator is not assigned to packages in different blocks
- Easily identify the block in which a component having a specific reference designator exists.

  This is helpful when you are debugging the design with respect to the board as you can trace back parts on the board to a specific block in the Design Editor

For example, if you have a hierarchical design named `MEMORY` with the two blocks `ROM` and `CONTROLLER`, you can assign the suffix `ROM` to the reference designators of all components in the `ROM` block and the suffix `CNTR` to the reference designators of all the components in the `CONTROLLER` block. The reference designators of components in the `ROM` block will be `U1_ROM`, `U2_ROM`, and so on. The reference designators of components in the `CONTROLLLER` block will be `U1_CNTR`, `U2_CNTR`, and so on. This unique identification of reference designators across blocks ensures that the same reference designator is not assigned to packages in different blocks. As a result, you can quickly debug a design with respect to the board layout by identifying the block in which a component having a specific reference designator exists.

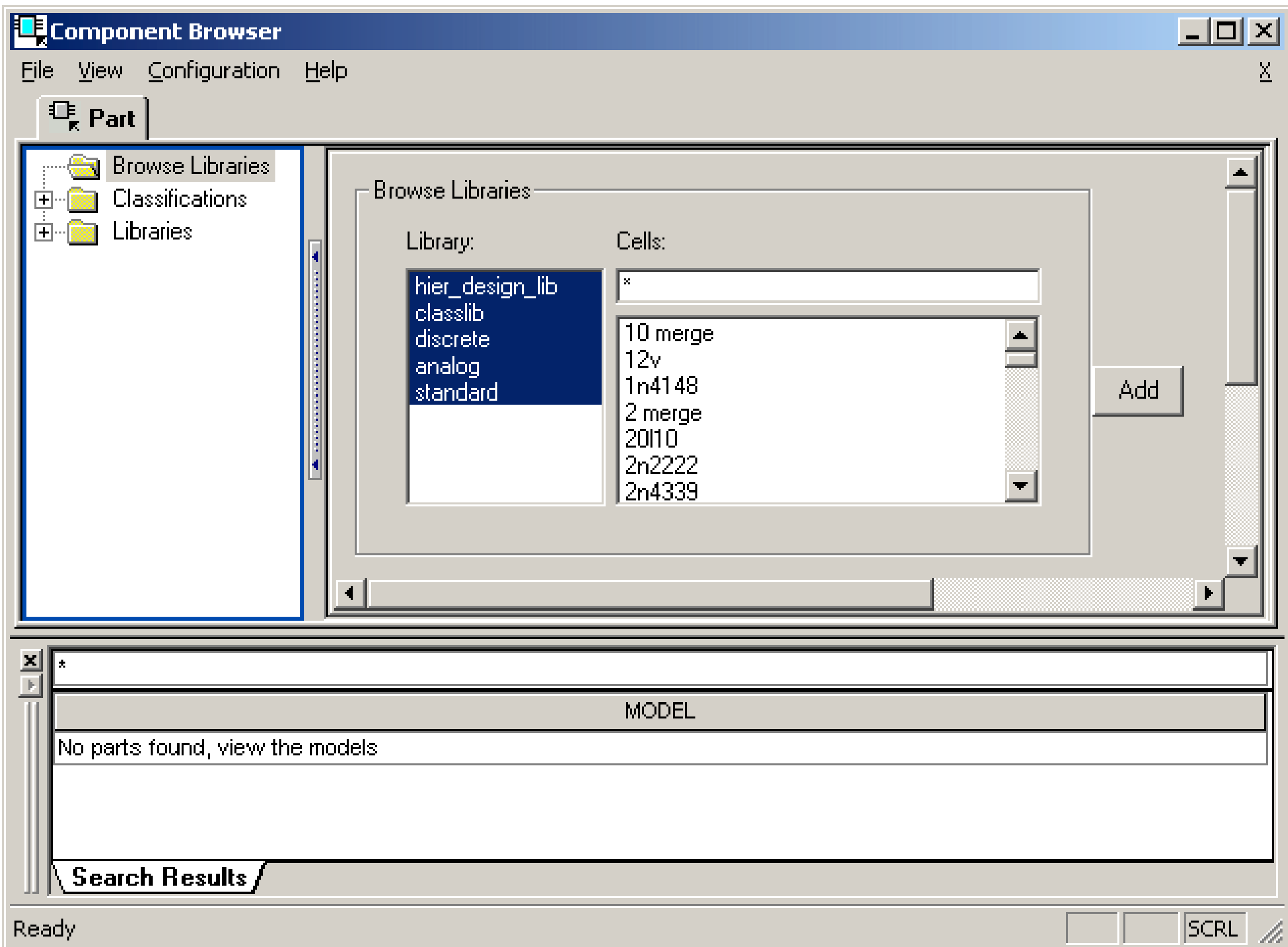
Procedure

1. Select the `cache` block in the Hierarchy Viewer.
2. Click the  *Descend* tool button.

3. Click the  *Add Component* tool button.

   The Component Browser window appears.

   Select all available libraries. For this, click the `hier_design_lib` (first) library and then keeping the `Shift` key pressed, click the `standard` (last) library.



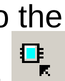
4. Type `act*` in the *Cells* field.
5. Select the `act574` component.
6. Select the row with the part name `ACT574` and the part number `ic345`.
7. Type 3 in the *Instances* field to specify that you will add three instances of the `ACT574` part.
8. Click the *Add* button.

Notice that three instances of the `ACT574` component are added to the `cache` block and appears in the Component List and in the Hierarchy Viewer.

9. Close the Component Browser.
10. Click *File - Save* to save the `cache` block.
11. Choose *File - Close* to close the `cache` block.

    In Lesson 6-1: Creating a Spreadsheet Block on page 218, you added an instance of the `cache` block in the `processor` block.

    You can now add another instance of the `cache` block with changed packaging options in the `processor` block. For example, you can define that reference designators for components in the new `cache` block should be any value between the range 30 and 39. Such a definition of reference designators may help you identify components across multiple instances of the same block.

12. Switch to the `processor` design by clicking its tab.
13. Click the  *Add Component* tool button.
14. Select the `cache` cell in the `hier_design_lib` library.
15. Click the *Add* button and close the Component Browser.

    The Block Packaging Options dialog box appears.

16. Select the *Use Ref. Des. Range* option button, enter `30 to 39` in the adjacent field, and click *OK*.

    A new instance of the `cache` block is added in the design.

17. Close the Component Browser.
18. Double-click the new instance of the `cache` block in the Hierarchy Viewer to open its tab.

    The `cache` design opens. Notice that the three instances of the `ACT574` part appear with the reference designators—`U30`, `U31`, and `U32`.

19. Close the tab for the `cache` block.

    If you view the Hierarchy viewer, you will see a two-level hierarchical design with the `processor` design at root level.

This design contains `cache`, `analog_io` and `clock` blocks. You have used the top-down methodology to create this design.

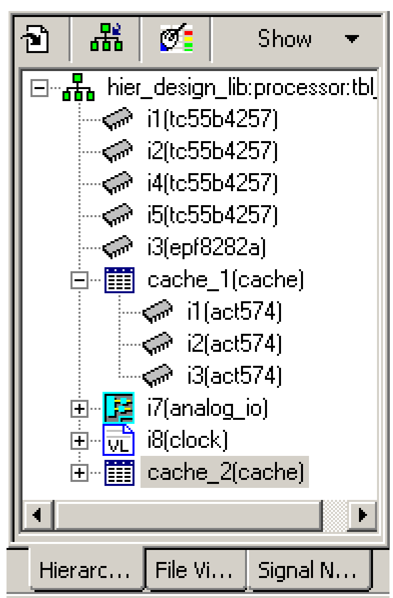
Notice that there are two instances of the `cache` block. Let's rename these instances as `cache_1` and `cache_2`.

20. Select the `i6` component from the Component List of the `processor` block.
21. Select *Design - Change - Name*.

    The instance name is selected and can be edited.

22. Type `cache_1` and press Enter.
23. Repeat the above steps to rename the other instance of the `cache` block to `cache_2`.

    The Hierarchy Viewer now displays the following design:



Summary

In this lesson, you learned to add components to a block and change their packaging options.

For More Information

See:

Working with Hierarchical Designs chapter of *Allegro Design Editor User Guide*.

Lesson 6-5: Editing Spreadsheet Blocks

Overview

In this lesson, you will learn to edit blocks in master and context mode.

Concept

You can edit the blocks in your design in the master mode or in the context of the root design.

- Master mode—In this mode, the changes you make to a block are applied to all instances of the block in the design. The block edited in the master mode can be used as a stand-alone block.

  The top-level or root design is always opened for editing in the master mode.

- Context mode—In this mode, the property and electrical constraint changes you make to a block are applied in context of the root design. However, the component and connectivity changes you make to the block are applied to all instances of the block in your design.

Procedure

1. Open the `cache_1` block in the context mode. For this, double-click the `cache_1` block in the Hierarchy Viewer.

   A new tab for the `cache` block opens. Notice that the titlebar displays the following text:

   `[In Context:processor.cache_1]`

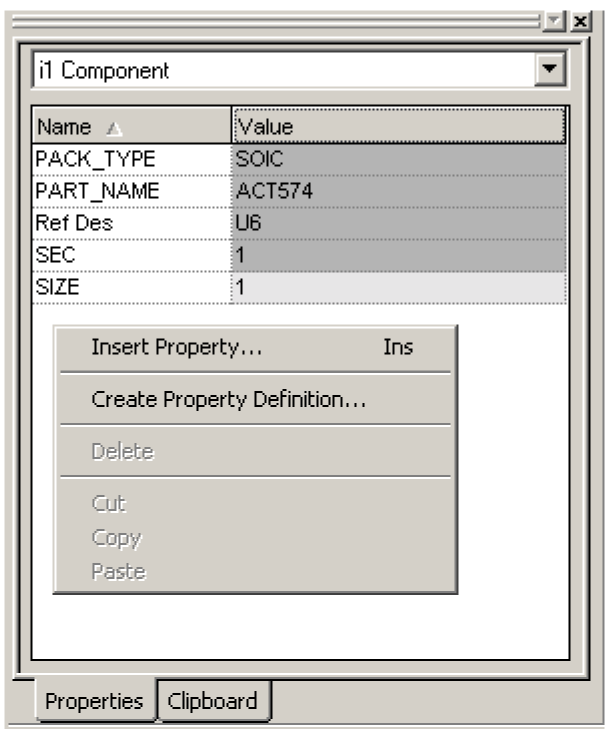
This indicates that the block with the block named cache_1 is open in the context of the design named processor.

2. Select the i1 instance in the Component List.
3. Connect the pins D<0..7> with the VD<0..7> IN port.
4. Connect the pins CLK with the VCLKA IN port.
5. Ground the OE pin. For this, connect the OE pin with the GND signal.

   The connectivity changes are applied to the cache design. These changes will appear in all instances of the cache block.

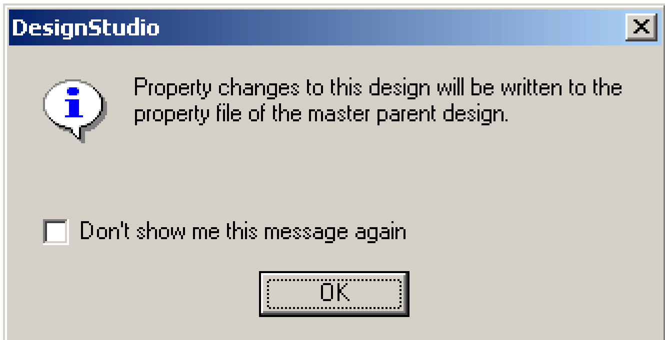
6. Select *View - Properties Window* to open the Properties window.
7. Select the i1 instance in the Component List.
8. Right-click in the Properties window to display the shortcut menu and select the *Insert Property* command.

   A new property row appears.



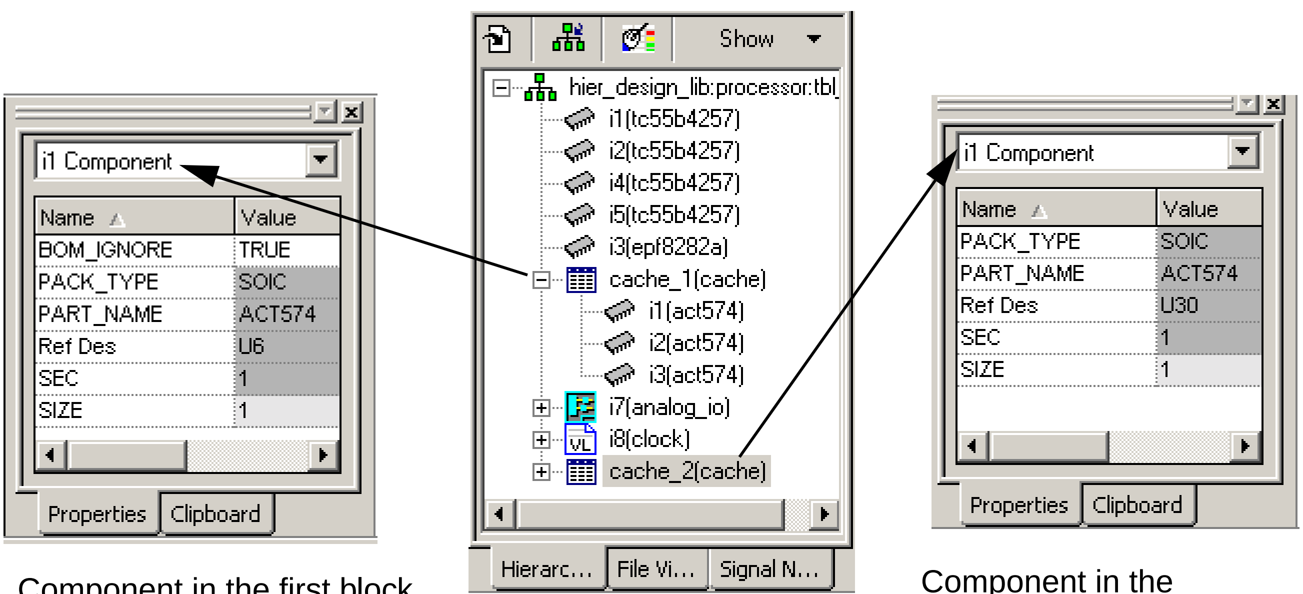
9. Specify the property name as BOM_IGNORE and the value as TRUE.

A message box stating that the property changes to this (`cache`) block will be written to the property file of the master parent (`processor`) design appears.



10. Click *OK*.
11. Select the `i1` component that has the `Ref Des` value `U30` in the `cache_2` block in the Hierarchy Viewer.

    Notice that the Properties Window refreshes.



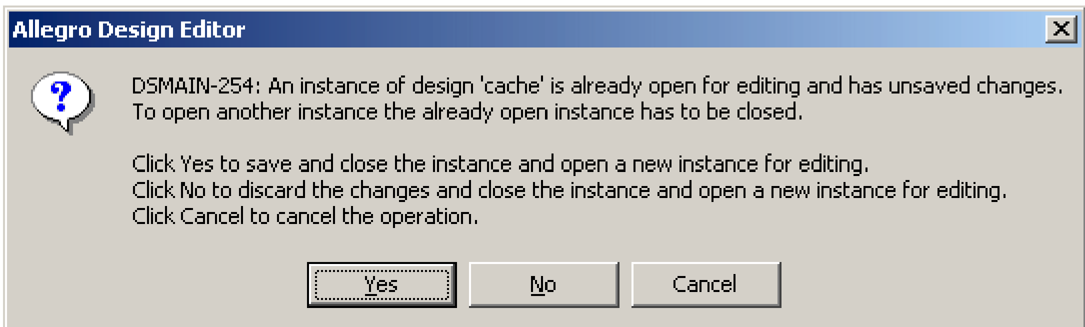
Component in the first block

Component in the second block

Notice that the Properties window does not show the `BOM_IGNORE` property you added in the `i1` component in the `cache_1` block. The property changes made in the context mode are applied to the property file of the master parent (`processor`) design and not in the property file of the `cache` block. As a result, only the impacted `cache` block instance in the master design `processor` is changed.

12. Select the `cache_2` block in the Hierarchy Viewer.
13. Click the  *Descend* tool button.

    A message box stating that you already have an instance (`cache_1`) open for editing appears. You are trying to open another instance of the same design block. The message prompts you to save or discard the changes.



14. Click *Yes* to save the changes.

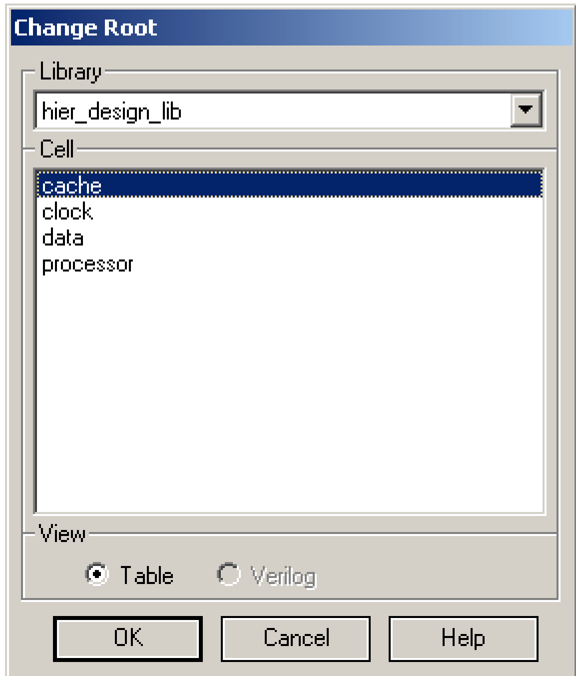
    A tab for the `cache_2` block opens. Notice that the connectivity changes you made for the component with the instance name `i1` in the other instance of the `cache` block are available in the component with the instance name `i1` of the current block. If you make connectivity changes for any component in a block in the Context mode, then the changes are always made in context of the root design. As a result, these connectivity changes appear in all instances of the block under the root design.

15. Assign the `BOM_IGNORE` property with the value `TRUE` to the component with the instance name `i2`.
16. Click *OK* to close the message box.
17. Press `Ctrl+Shift+S` or select *File - Save All* to save changes to all blocks.

    The `cache` block is saved, and the changes made to it will appear in context of the `processor` block. Changes in the `processor` design are also saved.

18. Select *Project - Change Root* for changing the root design for the project.

The Change Root dialog box appears.



19. Select the `cache` block and click *OK*.

    Notice that the root design changes to `cache`.

20. Select the `i1` component in the `cache` block.

    Notice that the Properties window refreshes and it does not show the `BOM_IGNORE` property you added in the `i1` component in the `cache_1` block in the Context mode earlier. The reason stems from the fact that the property changes made in the context mode are applied to the property file of the master parent (`processor`) design and not `cache` block.

21. Select the component with the instance name `i3` and assign the `BOM_IGNORE` property with value `TRUE` on it.

22. Select *Project - Change Root* for changing the root design back to `processor`.

    Note: If you get a message box to save unsaved designs, click *Yes*.

    The `processor` block appears as the root design.

23. Double-click the `cache_1` block in the Hierarchy Viewer.

    The `cache` block opens.

24. Select the `i3` instance of the `cache_1` block in the Hierarchy Viewer.

    The `BOM_IGNORE` property is listed as an assigned property in the Properties window. The value assigned to this property is `TRUE`.

25. Similarly, check the properties of the `i3` instance in the `cache_2` block.

    The `BOM_IGNORE` property with value `TRUE` is listed as an assigned property in the Property window for this instance too. Changes to properties made in master mode are available across all instances of the design.

Summary

In this lesson, you learned to edit blocks in master and context mode.

For More Information

See:

<u>Working with Hierarchical Designs</u> chapter of *Allegro Design Editor User Guide*.

Lesson 6-6: Creating a Third-Level Hierarchical Design

Overview

In this lesson, you will learn to create a block and add it in a midlevel design block.

Procedure

1. Select *Project - Change Root* and change the root design back to `cache`.

Note: If you get a message box to save unsaved designs, click *Yes*.

2. Choose *Design - Create Block*.

   The Create Block dialog box appears.

3. Type `cache_ctrl` in the *Block Name* field.

   The *Block Library* field shows that the new block being created will be added in the `hier_design_lib`. Retain this setting.

4. Click the *Add Ports* button.

5. To add an input port dq<7..0>, type dq<7..0> in the *Port Name* field and press `Tab` to move the *Port Type* field.

6. Type `IN` or select it from the *Port Type* drop-down list.

7. Repeat steps step 5 and step 6 to add the input ports, `gain`, `vclk`, and `vref,` and an output port named `out`.

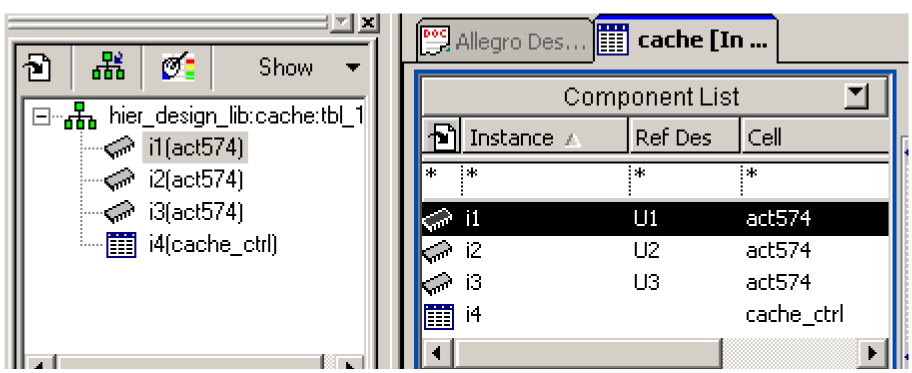
8. Select the *Add instance to design* check box to add an instance of the new block in the `cache` block.

9. Click *OK* to store the `cache_ctrl` cell in the `hier_design_lib` library.

   The Block Packaging Options dialog box appears.

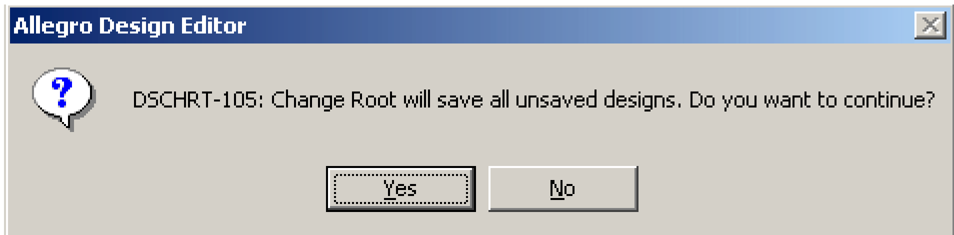
10. Click *OK* to accept the default packaging option.

    Notice that a new block named `cache_ctrl` appears as an instance in the Component List and also appears in the Hierarchy Viewer.



11. Select *Project - Change Root* for changing the root design back to `processor`.

A message box checking whether or not you want to save the design appears.



12. Click *Yes*.

    The `processor` block appears as the root design.

13. Double-click the `cache_1` block.

    The block expands and you can see the `cache_ctrl` block in it. You have a three level design hierarchy ready.

14. Select *File - Save All* to save the design.

Summary

In this lesson, you learned to create a three-level design hierarchy by creating a block and adding it in a mid-level design block. You have added this block using the top-down hierarchical design methodology.

For More Information

See:

Working with Hierarchical Designs chapter of *Allegro Design Editor User Guide*.

Lesson 6-7: Creating a Bottom-Up Hierarchical Design

Overview

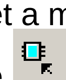
In this lesson, you will learn to create a bottom-up hierarchical design.

Concept

In the bottom-up methodology, you create a lower-level standalone block first and add it to a library. After a block is finalized, it can be integrated into the root design.

In this procedure, you will add a lower level block `data` as a sub-block in the `cache` block.

Procedure

1. Select *Project - Change Root* and change the root design back to `cache`.
2. If you get a message box to save unsaved designs, click *Yes*.
3. Click the  *Add Component* tool button.
4. Select all libraries and type `data` in the *Cells* field.
5. Select the `data` cell.
6. Click the *Add* button.

   The Block Packaging Options dialog box appears.
7. Select the *Use Ref. Des. Range* option button, type `40 to 49` in the adjacent field, and click *OK*.

   A new instance of the `data` block appears.
8. Close the Component browser.
9. Double-click to open the tab for the new `data` block instance.

   Notice that the three instances of the `tlc5602` part appear with the reference designators, `U40`, `U41`, and `U42`.
10. Close the `data` block.

    You have created a hierarchical design by adding it bottom-up.

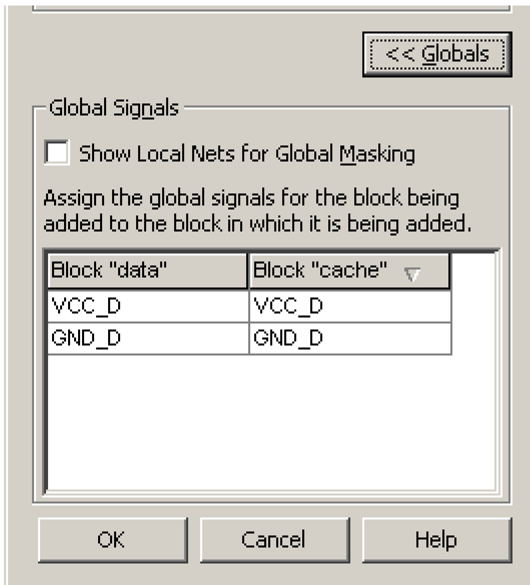
    Next, you will learn how to mask global signals.
11. Scroll the Signal List of the `cache` block.

    Notice that the signals `VCC_D` and `GND_D` appear in the Signal list of the `cache` block. These signals are global signals in the `data` block and are rippled up in the hierarchy.

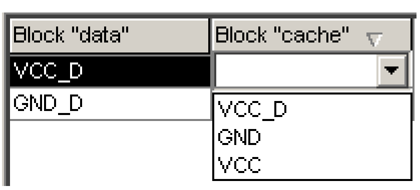
12. Select the `data` block in the Component List.
13. Select *Design - Block Packaging Options.*

    The Block Packaging Options dialog box appears.
14. Click the *Globals* button to display the *Global Signals* list.

    The Block Packaging Options dialog box expands to display the Global Signals list.



15. Alias the signal `VCC_D` in the `data` block with the signal `VCC` in the `cache` block. For this, select `VCC` in the cell in the same row as the `VCC_D` signal as shown below:



16. Similarly, alias the signal `GND_D` in the `data` block with the signal `GND` in the `cache` block.
17. Click *OK* to accept the changes in the Block Packaging Options dialog box.

    Notice that the signals `VCC_D` and `GND_D` are removed from the Signal List. Instead, the signals they are aliased to—`VCC` and `GND`—appear in the list.

18. Select *Project - Change Root* and set the root design as `processor`.

Summary

In this lesson, you learned to create a hierarchical design following the bottom-up design methodology. You also learned to mask global signals.

For More Information

See:

Working with Hierarchical Designs chapter of *Allegro Design Editor User Guide*.

Module 7: Generating Reports

Prerequisite

If you have not completed all the lessons in Module 6: Creating a Hierarchical Design, you must open the hier_design.cpm project located at <*your_work_area*>\modules\reports\hier_design in Design Editor and perform the steps described in this module.

For more information, see Understanding the Sample Design Files on page 14.

Lessons

This module consists of the following lessons:

- Lesson 7-1: Generating Standard Reports on page 250
- Lesson 7-2: Designing a Report Template on page 255
- Lesson 7-3: Customizing Existing Reports on page 262
- Lesson 7-4: Generating Block-Based Reports on page 266
- Lesson 7-5: Creating Cross-Tab Reports on page 271
- Lesson 7-6: Creating Custom Columns in Reports on page 276

Multimedia Demonstration

Click the link below to view a Flash-based multimedia demonstration of this module.

Generating Reports Using Allegro Design Editor

Completion Time

- 2 hours for written lessons

- 45 minutes for multimedia demonstrations

Lesson 7-1: Generating Standard Reports

Overview

Design Editor has some default report templates that are shipped with the product. In this lesson, you will learn how to use these templates to generate reports and how to select the output format.

Concept

Allegro Design Editor provides powerful report design and generation features. You can even design report templates and then use the templates to generate reports for any design.

The Generate Report dialog box is used to generate reports in Allegro Design Editor. To create a report, you need a template file. You can select an existing report template to create a report, or customize the report template on the fly or create a new template and use it to generate the report. An example of a report template is BOM (Bill of Materials) report.

You can generate reports in the following formats:

- HTML
- CSV
- Text in Tabular Format
- Editable Report File Format, which is an intermediate report format provided by Design Editor. Using this editor, you can customize column sizes, have custom columns, change fonts of cells and add custom rows in a BOM report as required.

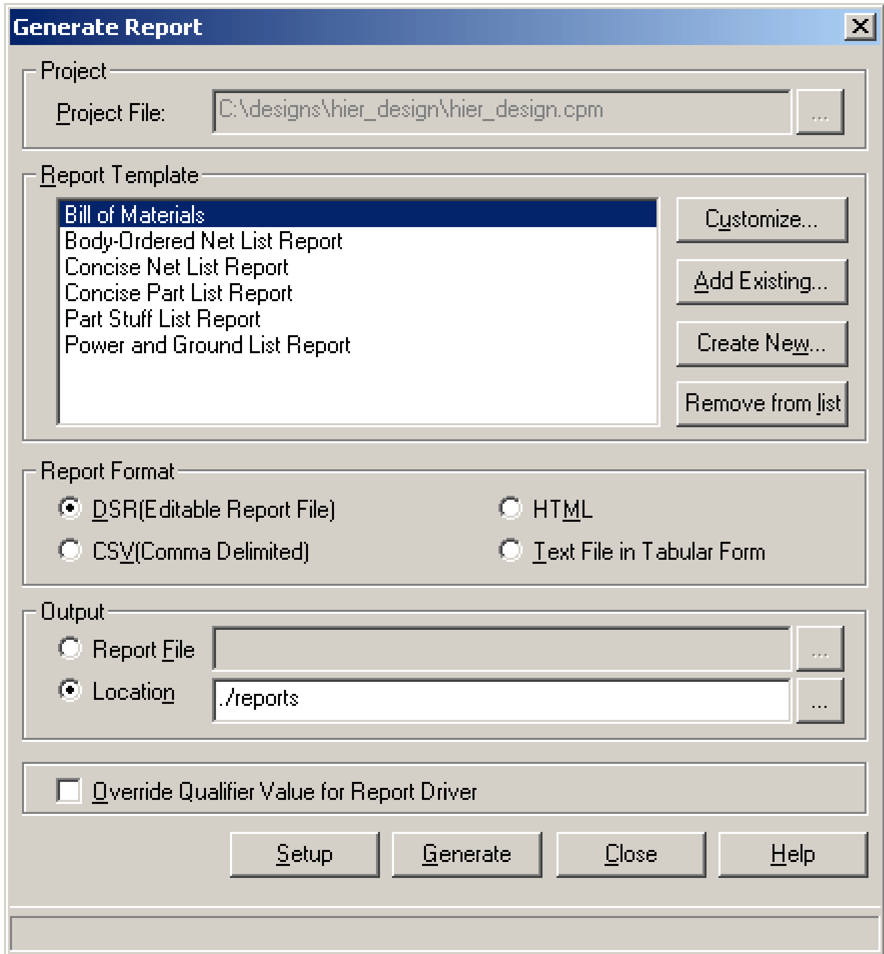
Procedure

Generating a Simple Report

1. Choose *Projects - Reports - Generate Reports*.

Note: You can run the `dsreportgen -proj <project_name>.cpm` command from the command line.

The Generate Report dialog box appears.



Notice that the dialog box has the name of project already seeded in.

2. Select the *Bill of Materials* in the *Report Template* box.
3. Select the *DSR (Editable Report File)* option button to specify the report format to be a custom editor within Allegro Design Editor.
4. Specify the output folder where the report will be located in the *Location* field.

   Note: The default report location is the `reports` folder under the root design.

5. Click *Generate* to generate the report.

Notice that a new tabbed page appears titled `processor_Bill of Materials.dsr`.

Allegro Des... | processor [... | **processor...**

	Physical Component	Ref Des	Quantity	Unit Cost	Total Cost
1	1N4148-BASE	SCH_CR1-SCH_CR	5	.23	5.75
2	2N2222-BASE	SCH_Q2	1	.43	0.43
3	2N4339-BASE	SCH_Q1	1	.49	0.49
4	ACT574_SOIC-BASE	U6-U8, U30-U32	6	0.65	23.4
5	CAP-.05UF	SCH_C1-SCH_C7	7	.18	8.82
6	CAP-0.1UF	C1, C2	2	.25	1
7	CAP-47UF	C3, C4	2	.22	0.88
8	CCADC-BASE	SCH_U16	1	12.77	12.77
9	CONN2-BASE	SCH_J8	1	1.98	1.98
10	DAC0800-BASE	SCH_U4	1	14.55	14.55
11	EPF8282A_PLCC-BASE	U5	1	1.05	1.05
12	GA1110-50-BASE	U1_V	1	26.44	26.44
13	LM102-BASE	SCH_U11	1	.89	0.89
14	LM3900-BASE	SCH_U8	1	1.10	1.1
15	MUX-BASE	SCH_U10	1	4.25	4.25
16	RESD-1K	SCH_R6, SCH_R8, SCH_R19	3	.29	2.61
17	RESD-1M	SCH_R1, SCH_R11, SCH_R13	3	.75	6.75
18	RESD-5.1M	SCH_R5, SCH_R7, SCH_R9, SCH_R10	4	.62	9.92
19	RESD-5K	SCH_R14, SCH_R1	2	.11	0.44
20	RESD-10K	R1-R4, SCH_R2, SCH_R4, SCH_R12	7	.35	17.15
21	RESD-10M	SCH_R16, SCH_R18,	4	.75	12
22	RESD-100K	SCH_R3, SCH_R17	2	.24	0.96
23	TC55B4257_SOIC-BAS	U1-U4	4	3.65	58.4
24	TLC5602_SOIC-BASE	U33-U35, U40-U42	6	3.25	117
25	TRANS-BASE	SCH_U12	1	2.25	2.25
TOTAL					331.28

Notice that the report has as many rows as the number of physical components in designs. There are five columns listing `Physical Component`, `Ref Des`, `Quantity`, `Unit Cost` and `Total Cost.` Depending upon the report template selected, the number of rows and columns displayed will change.

Note: The specific data about components

6. Select *File - Close* to close the tab for the BOM report.

Changing Report Formats

1. Choose *Projects - Reports - Generate Reports*.

2. Select *HTML* in the *Report Template* box.

3. Click *Generate*.

Notice that a new tabbed page appears displaying the html report. This report is displayed in a tabbed page titled `processor_Bill of Materials.html` file.

	Physical Component	Ref Des	Quantity	Unit Cost	Total Cost
1	1N4148-BASE	SCH_CR1-SCH_CR5	5	.23	5.75
2	2N2222-BASE	SCH_Q2	1	.43	0.43
3	2N4339-BASE	SCH_Q1	1	.49	0.49
4	ACT574_SOIC-BASE	U6-U8, U30-U32	6	0.65	23.4
5	CAP-.05UF	SCH_C1-SCH_C7	7	.18	8.82
6	CAP-0.1UF	C1, C2	2	.25	1
7	CAP-47UF	C3, C4	2	.22	0.88
8	CCADC-BASE	SCH_U16	1	12.77	12.77
9	CONN2-BASE	SCH_J8	1	1.98	1.98
10	DAC0800-BASE	SCH_U4	1	14.55	14.55
11	EPF8282A_PLCC-BASE	U5	1	1.05	1.05
12	GA1110-50-BASE	U1_V	1	26.44	26.44
13	LM102-BASE	SCH_U11	1	.89	0.89
14	LM3900-BASE	SCH_U8	1	1.10	1.1
15	MUX-BASE	SCH_U10	1	4.25	4.25
16	RESD-1K	SCH_R6, SCH_R8, SCH_R19	3	.29	2.61
		SCH_R1,			

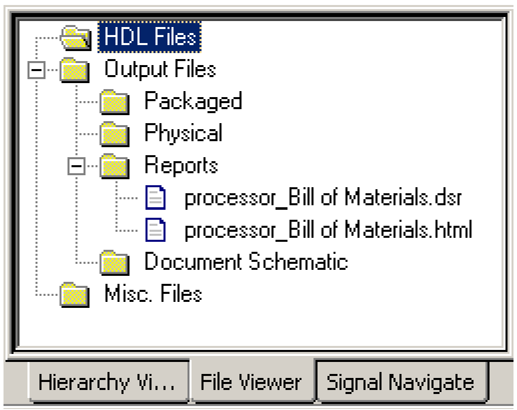
You can repeat the above steps to generate a CSV format report. The CSV report is displayed in a tabbed page titled `processor_Bill of Materials.csv`. You can open this file in Microsoft Excel.

4. Select *File - Close* to close the tab for the BOM report.

Viewing Reports from the File Viewer

1. Choose *View - File Viewer.*

   The File Viewer appears. The File Viewer displays the files related to your design and lets you open the files from Design Editor.



2. Double-click `processor_Bill of Materials.dsr`.

   The selected report opens in a new tab. The information displayed in this tab is same as one generated in step 5 of Generating a Simple Report section.

3. Select *File - Close* to close the tab for the BOM report.

Summary

In this lesson, you learned to create reports for your design. You can create a standard report in different file formats. You can format reports created in the `.dsr` format. You can add new columns or rows in generated reports.

For More Information

See:

Creating Reports chapter of *Allegro Design Editor User Guide*.

Lesson 7-2: Designing a Report Template

Overview

In this lesson, you will learn to create a report template and use it to generate reports.

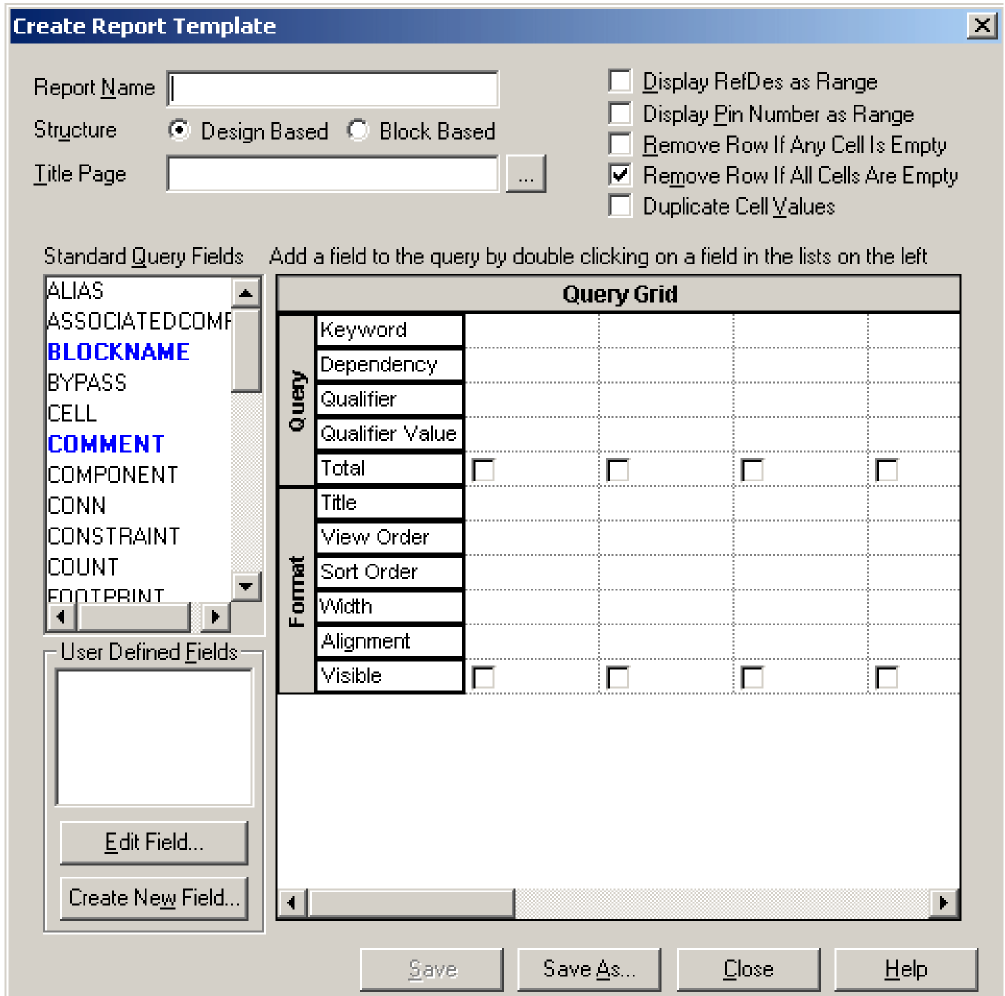
Concept

Allegro Design Editor uses report template (`.tpt`) files to create reports. You can create a new report template file or customize an existing report template from the Generate Report dialog box. A custom report allows you to define the order of keywords that will generate the query. You can change the view order, sorting order, visibility, alignment, width and other such characteristics in the report.

Procedure

1. Choose *Projects - Reports - Generate Reports*.
2. Click the *Create New* button.

The Create Report Template dialog box appears.



3. Type `Associated Component Report` to define the *Report Name*.

4. Select the *Design Based* option button to define that the report template should create reports for the entire design.

   You can select a text file as the title for your report. The text in this file would be displayed as the title page for the report.

5. Browse to select the *Title.txt* file in the project directory in the *Title Page* field.

6. Select the *Remove Row if Any Cell is Empty* check box to delete those rows in the report that have any cell as empty.

7. Clear the *Duplicate Cell Values* check box to ensure that duplicate values are not separately listed in the report.

   It's now time to define the query grid. The query grid displays the fields that can be used to generate data for the report. These fields can be added as keywords in the report. The order of keywords in the Query Grid define how the report will appear.

8. Select the first keyword as `REFDES`. To select the keyword, click in the first cell next to the *Keyword* cell and choose `REFDES` from the list box.

   Notice that a default title `Ref Des` is assigned. This title will be used as the column heading in the BOM report for the `REFDES` keyword. You may like to customize the default title.

9. Type `Component Reference Designator` in the *Title* cell for the `REFDES` column.

10. Select the second keyword as `PINNUM`. This time, try the second method of entering keywords. Select `PINNUM` in the *Standard Query Fields* and drop it in the second cell next to the *Keyword* cell.

    Notice that the *Dependency* field for the `PINNUM` column is automatically set as `REFDES` and the title is set as `Pin Number`. When you add a keyword in the query grid, another keyword you already added in grid may be automatically set as the dependency for the keyword. Since the keyword `REFDES` was added in the query grid in <u>step 8</u>, when you add the `PINNUM` keyword in the query grid in <u>step 10</u>, `REFDES` is automatically set as the dependency for the `PINNUM`. If there are multiple keywords already set, then Design Editor uses a pre-defined fixed order to calculate which keyword should be set as dependency. You can select another keyword as the dependency for the `PINNUM` keyword.

11. Select the third keyword as `ASSOCIATEDCOMP`. This time, try the third method of entering keywords. Type `AS` in the third cell next to the *Keyword* cell and the word `ASSOCIATEDCOMP` appears. Press `Enter`.

    Notice that the *Dependency* field for the `ASSOCIATEDCOMP` column is automatically set as `REFDES`.

12. Type `Associated Component Type` in the *Title* cell for the `ASSOCIATEDCOMP` column.

13. Select the fourth keyword as `REFDES`.

    Notice that the *Dependency* field for the `REFDES` column is automatically set as `PINNUM` and the title is set as `Ref_Des_1`.

14. Type `Associated Component Reference Designator` in the *Title* cell for the `REFDES` column.

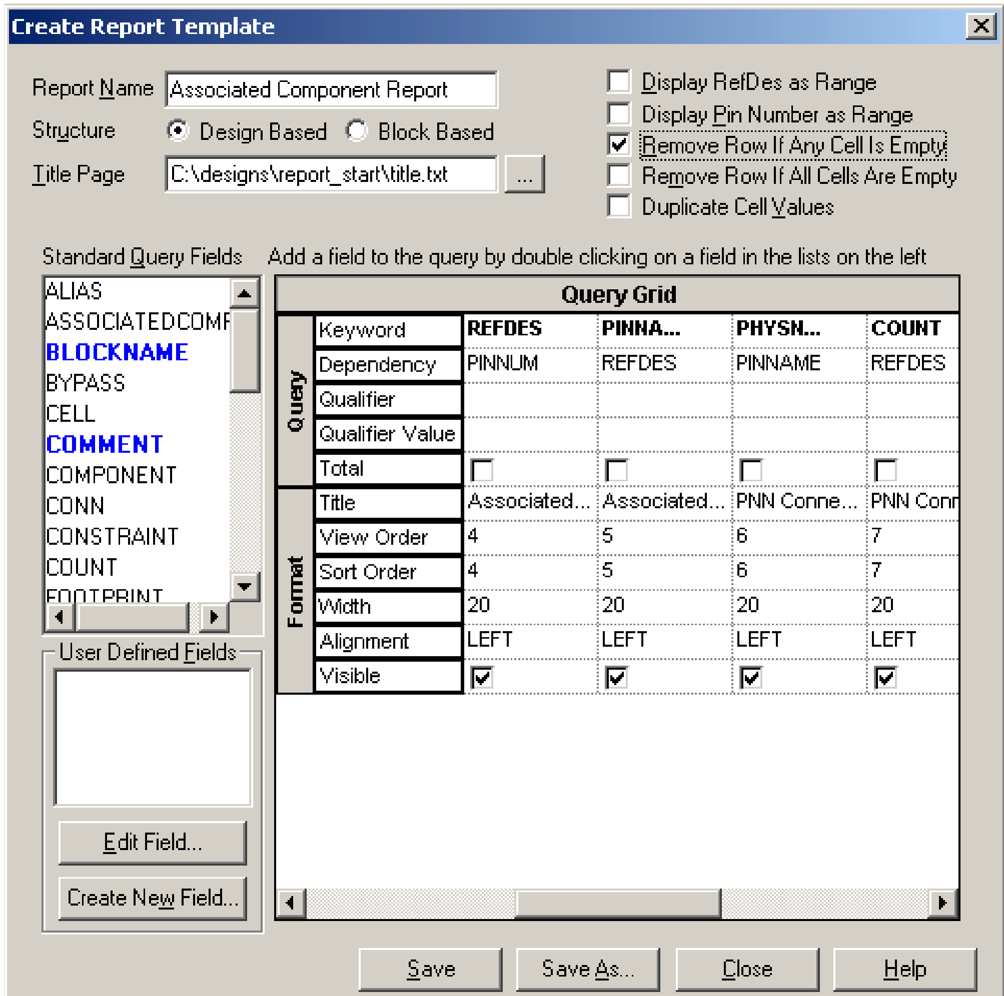
15. Assign `PINNAME` as the keyword in next column and change its title to `Associated Component Pin Name`.

16. Assign `PHYSNET` as the keyword in next column and change its title to `PNN Connected to the Associated Component Pin`.

    Notice that the dependency of the `PHYSNET` column is set to `PINNAME`.

17. Assign `COUNT` as the keyword in next column, change the dependency to `ASSOCIATEDCOMP`, and set the title of the column to `PNN Connected to the Associated Component Count`.

The Create Report template Dialog box should look like the figure shown below:



18. Click the *Save As* button to save the report with a different name.

19. Enter `AssocCompReport.tpt` in the *File Name* field and click *Save.*

20. Click *Close* to close the Create Report template dialog box.

Notice that the `Associated Component Report` is selected in the *Report Template* box.

Tip

If a report template is created but is not listed in the *Report Template* box, use the *Existing* button to include the template file in the available report templates list.

21. Select the *DSR (Editable Report File)* option button to specify the report format.

22. Specify the output folder where the report will be located in the *Location* field.

23. Click the *Generate* button to generate the report.

The Associated Component Report is generated.

Notice that the name of the report is `processor_Associated Component Report.dsr`. At the top of the report, there is a two line text entry stating:

```
XYZ Firm - ABC Department
# This is the header file that will be used as Title page
in reports.
```

The report lists 7 columns. If you have components associated with other components, then for those components all seven columns will show value. Otherwise, the component reference designator and pin numbers would be listed.

	Component Reference Designator	Pin Number	Associated Component Type	Associated Component Reference Designator	Associated Component Pin Name	PNN Connected to the Associated Component Pin	PNN Connected to the Associated Component Count
	XYZ Firm - ABC Department # This is the header file that will be used as Title page in reports.						
1	C1	1		C1	a	VCC	1
2					b	GND	
3		2		C1	a	VCC	1
4					b	GND	
5	C2	1		C2	a	VCC	1
6					b	GND	
7		2		C2	a	VCC	1
8					b	GND	
9	C3	1		C3	a	VCC	1
10					b	GND	
11		2		C3	a	VCC	1
12					b	GND	
13	C4	1		C4	a	VCC	1
14					b	GND	
15		2		C4	a	VCC	1
16					b	GND	
17	R1	1		R1	a	GND	1
18			Pullup		b	RCS0	
19		2		R1	a	GND	1
20			Pullup		b	RCS0	

Notice that duplicate component reference designators are listed only once. If you want the component reference designator to be repeated for each pin number, select the *Duplicate Cell Values* check box in the `AssocCompReport.tpt`.

Summary

In this lesson, you learned to create report templates and use them to generate reports for your design.

For More Information

See:

Creating Reports chapter of *Allegro Design Editor User Guide*.

Lesson 7-3: Customizing Existing Reports

Overview

In this lesson, you will learn to customize an existing report template and use it to generate reports. You may have standard report templates available for your firm, and may want to customize one of these report templates to quickly generate a report you need.

Concept

Allegro Design Editor uses report template (`.tpt`) files to create reports. You can open an existing report template and change its parameters or use it to create new report template.

Procedure

Customizing Report Templates

1. Choose *Projects - Reports - Generate Reports*.
2. Select *Bill of Materials* in the *Report Template* field.
3. Click the *Customize* button.

   The Create Report template dialog box appears. Notice that the *Total* check box in the *Count* column is not selected. This is the reason why the BOM report generated in the *Generating a Simple Report* on page 250 is not very effective as the total number of components is not displayed. Let's modify the BOM template to show the count of components.
4. Select the *Total* check box for the *Count* column.
5. Type `BOM with Component Count` in the *Report Name* field.
6. Click *Save As* and save the report with the name `BOM_count.tpt`.

Tip

If you want to create a report template and want it to be available for use across your organization, you can save it in the location <*`your_inst_dir`*>`/share/cdssetup/tdd/custom_report_templates`.

7. Click *Close* to close the Create Report template dialog box.

8. Select the `BOM with Component Count` report in the *Report Template* box.

9. Click the *Generate* button to generate the report.

The report is generated. Notice that the total component count is now displayed.

	Physical Component	Ref Des	BOM with Component Count	Unit Cost	Total Cost
5	CAP-.05UF	SCH_C1-SCH_C7	7	.18	8.82
6	CAP-0.1UF	C1, C2	2	.25	1
7	CAP-47UF	C3, C4	2	.22	0.88
8	CCADC-BASE	SCH_U16	1	12.77	12.77
9	CONN2-BASE	SCH_J8	1	1.98	1.98
10	DAC0800-BASE	SCH_U4	1	14.55	14.55
11	EPF8282A_PLCC-BASE	U5	1	1.05	1.05
12	GA1110-50-BASE	U1_V	1	26.44	26.44
13	LM102-BASE	SCH_U11	1	.89	0.89
14	LM3900-BASE	SCH_U8	1	1.10	1.1
15	MUX-BASE	SCH_U10	1	4.25	4.25
16	RESD-1K	SCH_R6, SCH_R8, SCH_R19	3	.29	2.61
17	RESD-1M	SCH_R1, SCH_R11, SCH_R13	3	.75	6.75
18	RESD-5.1M	SCH_R5, SCH_R7, SCH_R9, SCH_R10	4	.62	9.92
19	RESD-5K	SCH_R14, SCH_R15	2	.11	0.44
20	RESD-10K	R1-R4, SCH_R2, SCH_R4, SCH_R12	7	.35	17.15
21	RESD-10M	SCH_R16, SCH_R18, SCH_R20, SCH_R21	4	.75	12
22	RESD-100K	SCH_R3, SCH_R17	2	.24	0.96
23	TC55B4257_SOIC-BAS	U1-U4	4	3.65	58.4
24	TLC5602_SOIC-BASE	U33-U35, U40-U42	6	3.25	117
25	TRANS-BASE	SCH_U12	1	2.25	2.25
TOTAL			68		331.28

Customizing Report Templates to Show Specific Values

You can customize an existing report to display components or properties of specific values. In the following example, you will select the `Concise_net_list.tpt` template and customize it to display netlist for components with `RefDes U1*`.

1. Choose *Projects - Reports - Generate Reports*.
2. Select *Concise Net List Report* in the *Report Template* field.
3. Click the *Customize* button.
4. Type *Concise Net List Report for Components with RefDes Beginning with Letter U* in the *Report Name* field.
5. Select *Value* in the *Qualifier* cell of the *RefDes* column.
6. Enter `U1*` as the value in the he *Qualifier Value* field of the *RefDes* column.
7. Click *Save As* and save the template with name `Net_list_U_star.tpt`.
8. Click *Close* to close the Create Report template dialog box.
9. Select the `Concise Net List for U1*` report in the *Report Template* box.
10. Click the *Generate* button to generate the report.

The report is generated.

Allegro Des... | processor [... | processor...

	Physical Net Name	Ref Des	Pin Number	Physical Component
53	N12V		1	
54	N12V		2	
55	N12V		6	
56	N12V		7	
57	NCS		11	
58	OE		32	
59	Q0_V	U1_V	17	GA1110-50-BASE
60	Q1_V	U1_V	20	GA1110-50-BASE
61	Q2_V	U1_V	21	GA1110-50-BASE
62	Q3_V	U1_V	22	GA1110-50-BASE
63	Q4_V	U1_V	3	GA1110-50-BASE
64	Q5_V	U1_V	24	GA1110-50-BASE
65	RA<0>	U1	5	TC55B4257_SOIC-BASE
66	RA<0>		76	
67	RA<1>	U1	4	TC55B4257_SOIC-BASE
68	RA<1>		5	
69	RA<1>		71	
70	RA<2>	U1	3	TC55B4257_SOIC-BASE
71	RA<2>		70	
72	RA<3>	U1	2	TC55B4257_SOIC-BASE
73	RA<3>		69	
74	RA<4>	U1	32	TC55B4257_SOIC-BASE
75	RA<4>		67	
76	RA<5>	U1	31	TC55B4257_SOIC-BASE
77	RA<5>		66	
78	RA<6>	U1	30	TC55B4257_SOIC-BASE

Notice that the report lists the physical net names and pin numbers for the components with `RefDes` beginning with letters `U1`. If you scroll down the list, you will find over 200 components have reference designators starting with the letter `U1`.

Summary

In this lesson, you learned to customize an existing report template and use it to generate reports.

For More Information

See:

Creating Reports chapter of *Allegro Design Editor User Guide*.

Lesson 7-4: Generating Block-Based Reports

Overview

In this lesson, you will learn to generate reports for different blocks. These reports are sorted by blocks in a design.

Concept

Allegro Design Editor creates two types of reports—block-based report and design-based report. You have learned creating design-based reports, which report on components across the entire design. In this lesson, you will learn to create a template for block-based report and use it to generate reports sorted on basis of all blocks in the design.

Procedure

Creating the Template for Block-based Report

1. Choose *Projects - Reports - Create Template*.

   The Create Report Template dialog box appears.

2. Type `Block Based Net Report` to define the *Report Name*.
3. Select the *Block Based* option button to define that the report template should sort the report by blocks in the design.
4. Select the *Remove Row if Any Cell is Empty* check box to delete those rows in the report that have any cell as empty.
5. Clear the *Duplicate Cell Values* check box to ensure that duplicate values are not separately listed in the report.

   You can now define the query grid.

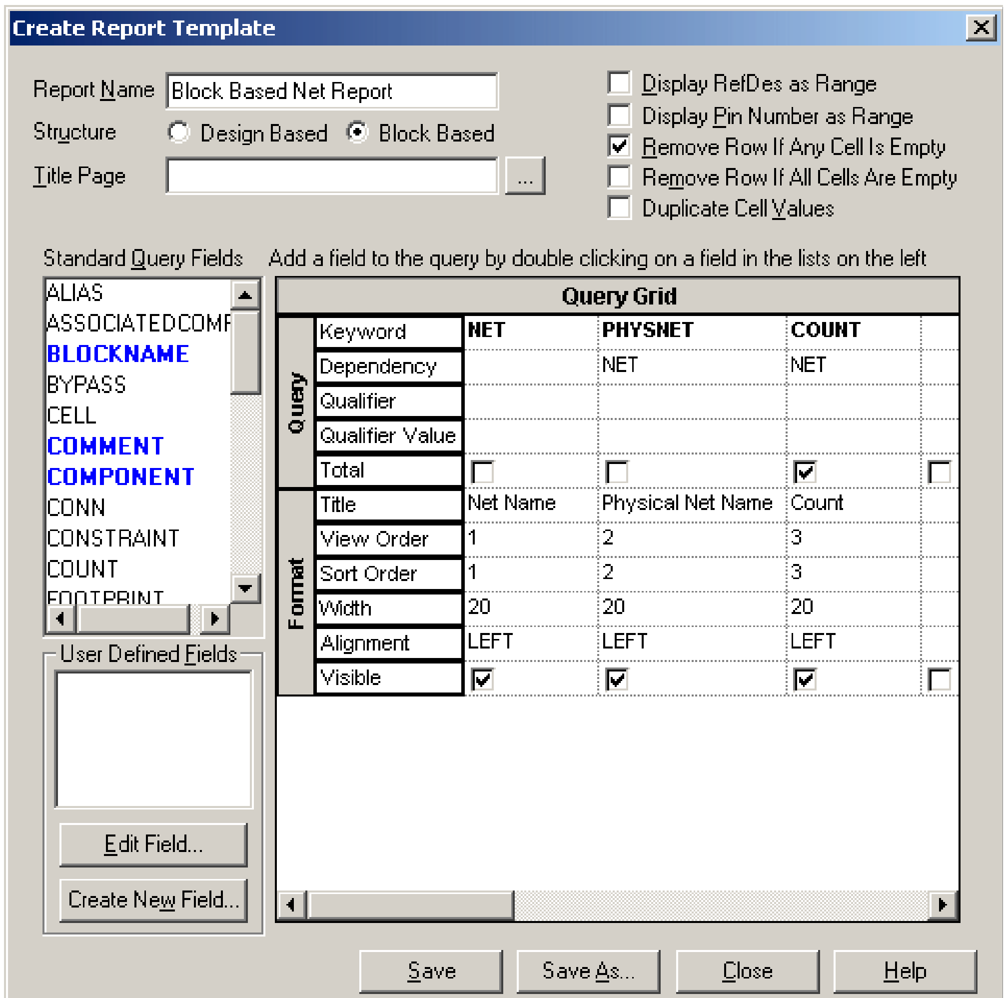
6. Select first keyword as `Net`.
7. Type `Net Name` in the *Title* cell for the `Net` column.

8. Select second keyword as `PHYSNET` and specify its title as `Physical Net Name`.

   Notice that the dependency of the `PHYSNET` column is set as `NET`.

9. Select third keyword as `Count` and select its *Total* field.

   The Create Report template Dialog box should look like the figure shown below:



10. Click the *Save As* button.

11. Select the `report_start/temp` directory and save the file with the name `BlockBasedNetReport.tpt`.

12. Click *Close* to close the Create Report template dialog box.

Generating the Report

1. Choose *Projects - Reports - Generate Reports*.

   You need to select the custom report you created. All reports created in the root directory automatically appear in the *Report Template* list. To get another report template not appearing in the default path, use the *Add Existing* button.

2. Click the *Add Existing* button.

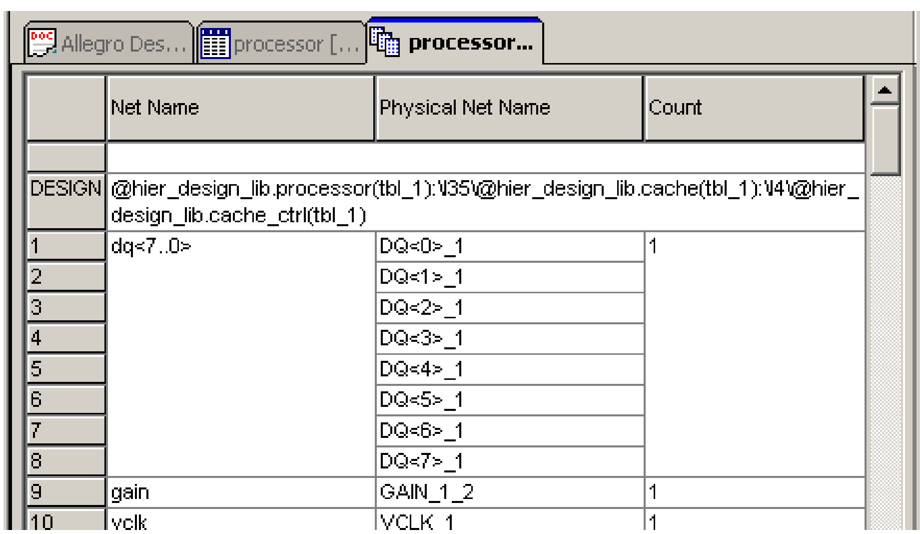
3. Open the `BlockBasedNetReport.tpt` report template from the `hier_design/temp` directory.

   Notice that the `Block Based Net Report` is selected in the *Report Template* box.

4. Select the *DSR(Editable Report File)* option button to specify the report format.

5. Click the *Generate* button to generate the report.

   The report lists nets available at different design levels, along with their physical net names and connection count of each net.



	Net Name	Physical Net Name	Count
DESIGN	@hier_design_lib.processor(tbl_1):\I35\@hier_design_lib.cache(tbl_1):\I4\@hier_design_lib.cache_ctrl(tbl_1)		
1	dq<7..0>	DQ<0>_1	1
2		DQ<1>_1	
3		DQ<2>_1	
4		DQ<3>_1	
5		DQ<4>_1	
6		DQ<5>_1	
7		DQ<6>_1	
8		DQ<7>_1	
9	gain	GAIN_1_2	1
10	vclk	VCLK_1	1

As you scroll down the list, you will see BOM information for different design blocks.

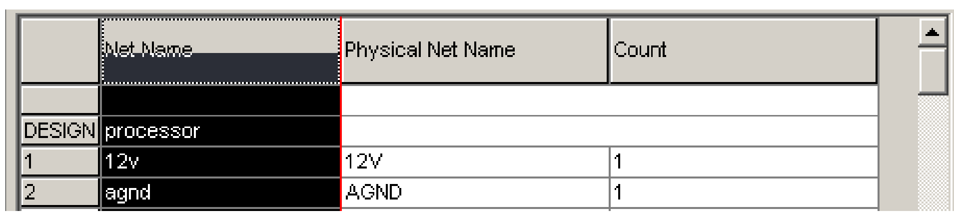
Editing the .dsr Report

You can edit the displayed report in multiple ways. In next few steps, you will learn few of these methods.

1. To change the order of columns, select a column and drag it to the place you wish. For example, try this.

   a. Click the *Net Name* column header to select it.

   Keeping the left mouse button pressed, drag the column to the left. As you drag the column, notice that a rectangle appears indicating a drag operation. As you pass a column separator, a red line appears indicating the current position. Drop the *Net Name* column after the *Physical Net Name* column.

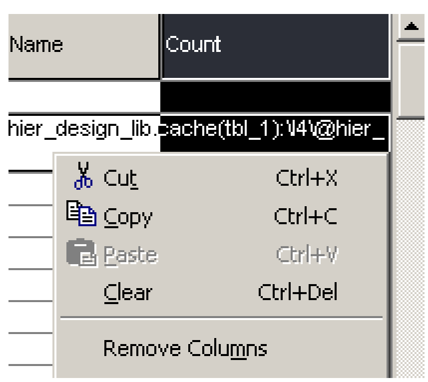


   Note: You can drag and drop multiple rows or columns to change their display order.

2. To delete a column, for example the *Count* column:

   a. Select the *Count* column header and right-click to display the short-cut menu.

   b. Select the *Remove Columns* command.



3. To quickly select a component, for example the component with the `Ref Des` `U11`:

a. Select the signal with the net name `dq<7..0>` and the physical net name `DQ<7>_1`.

b. Select *Edit - Highlight*.

The component with `Ref Des U11` is highlighted in the design.

4. To quickly change the font and alignment; for example, to center align a row and apply bold style, select the cells that require alignment change, and choose:

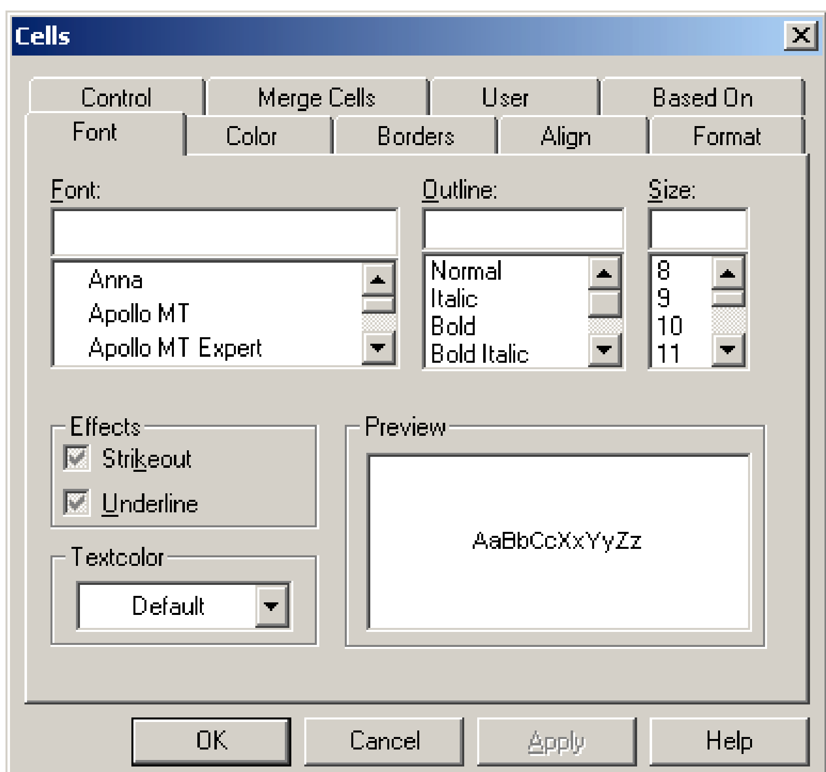
   a. *Format - Align - Center.*

   b. *Format - Style - Bold.*

5. To change cell level format:

   a. Select the cell and choose *Format - Cells*.

   The Cells dialog box appears.



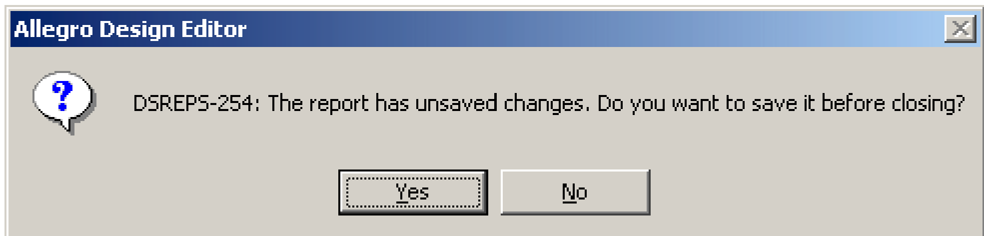
   b. To change a font, select a new font, outline or cell.

c. To change colors, select the *Color* tabbed page and select foreground or background color, shade and if required a 3-D effect. You can preview the selection before you apply it.

d. Click *Ok* to accept the changes.

6. Close the report by selecting *File - Close*.

   You may get the message that report has unsaved changes and whether you would wish to save those changes.



7. Click *Yes* to save the changes.

Summary

In this lesson, you learned to generate reports that are sorted with blocks. You also learned how to edit a `.dsr` report.

For More Information

See:

Creating Reports chapter of *Allegro Design Editor User Guide*.

Lesson 7-5: Creating Cross-Tab Reports

Overview

In this lesson, you will generate a cross tab report. A cross tab report can provide information such as details of which pin names of a given component instance is connected to which signal.

Concept

You can enter the name of another keyword you have added in the query grid as the title for a keyword to generate a cross-tab report. The title should be entered as `%keyword_name%` where *`keyword_name`* is the name of another keyword in the query grid.

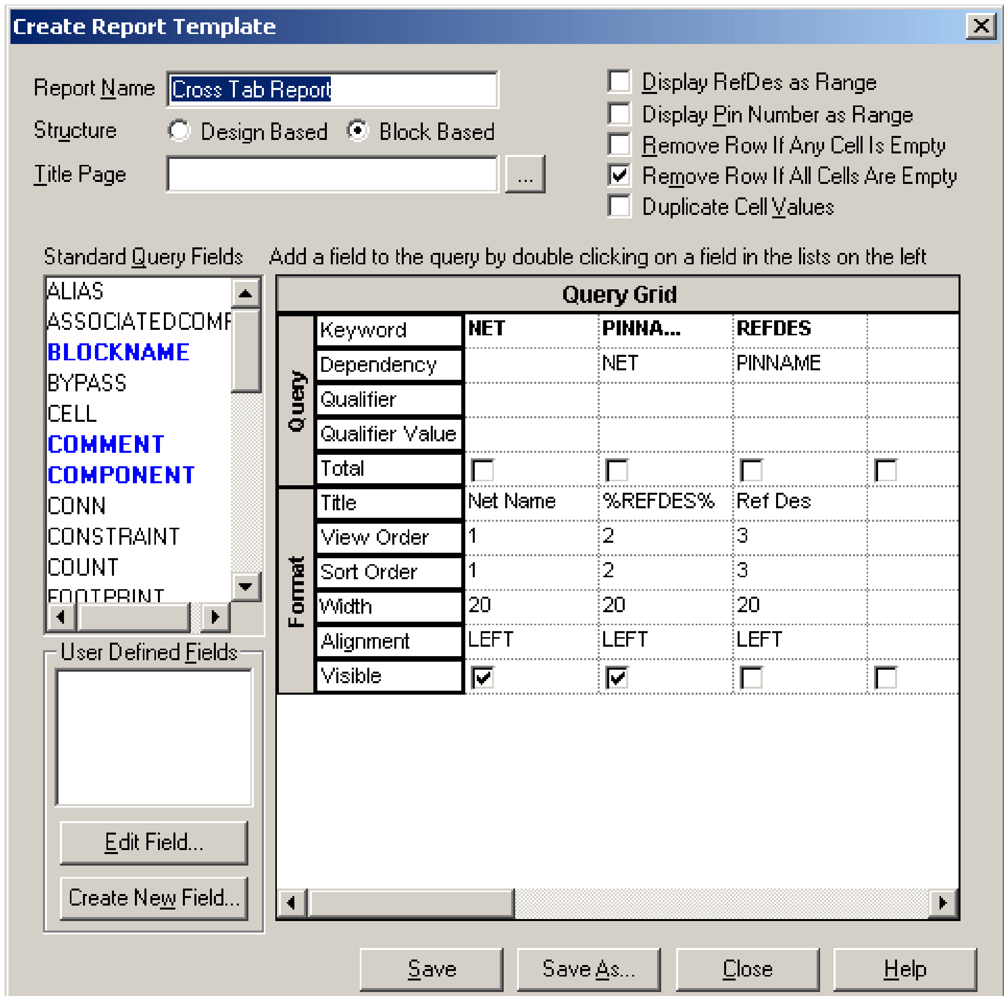
When you generate the report, the value of the keyword you entered as the title will be displayed as the column heading in the report. For example, if you enter `%keyword_name%` as the title for the keyword `PINNUM`, the reference designators of components in the design are displayed as column headings in the report and the pin numbers of a given reference designator appear in the column for the reference designator.

Procedure

1. Choose *Projects - Reports - Create Template*.
2. Type `Cross Tab Report` to define the *Report Name*.
3. Select the *Block Based* option button to define that the report template should sort the report by blocks in the design.
4. Select the *Remove Row if Any Cell is Empty* check box to delete those rows in the report that have any cell as empty.
5. Clear the *Duplicate Cell Values* check box to ensure that duplicate values are not separately listed in the report.
6. Select first keyword as `Net`.
7. Type `Net Name` in the *Title* cell for the `Net` column.
8. Select second keyword as `PINNAME`.

   Notice that the dependency for the `PINNAME` is automatically set as `NET`.
9. Select third keyword as `REFDES`.
10. Specify the title of the `PINNAME` column as `%REFDES%`.
11. Clear the *Visible* check box of the `REFDES` column.

The Create Report template Dialog box should look like the figure shown below:



12. Click the *Save As* button and save the report with the name `CrossTabReport.tpt`.

13. Click the *Close* button to close the Create Template dialog box.

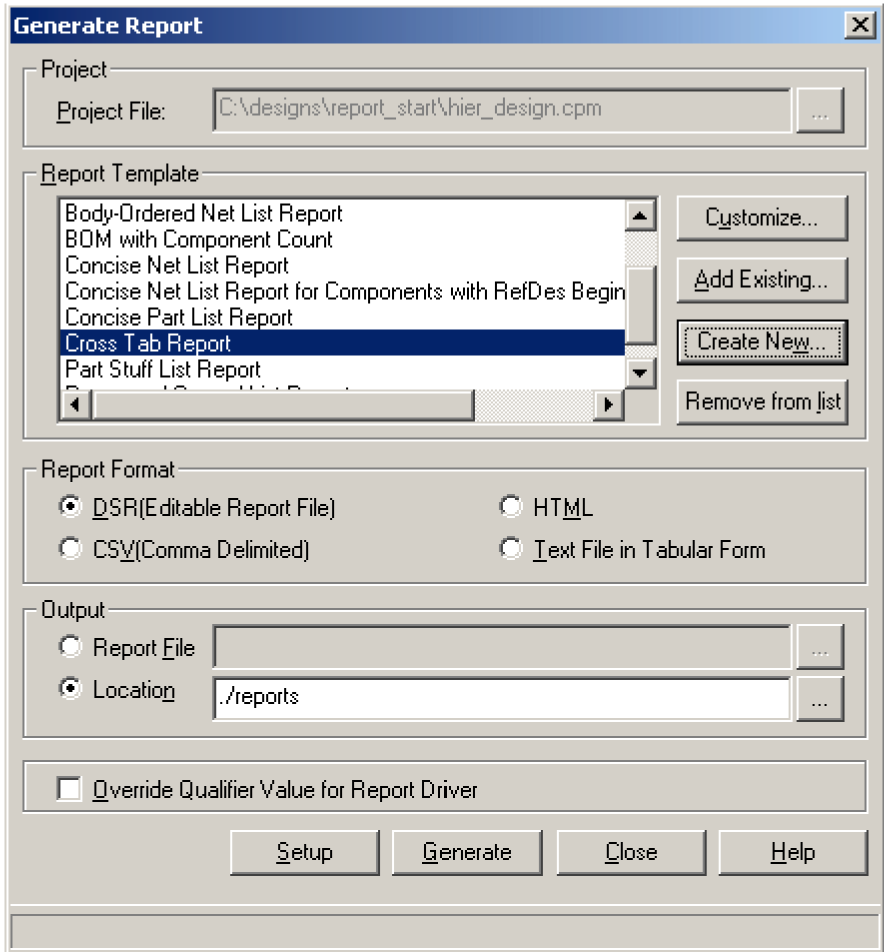
You have created the template file for generating a cross-tab report. You can now use this template to generate the cross-tab report.

14. Choose *Projects - Reports - Generate Reports*.

15. Select the *Cross Tab Report* in the *Report Template* box.

16. Select the *DSR(Editable Report File)* option button to specify the report format.

The Generate Report dialog box should have the following settings:



17. Click the *Generate* button to generate the report.

Notice that the report lists all net names along with the pin names of components where they are connected. You can scroll down (and/or right) the report to see the details.

	Net Name	C4	C6	C7	SCH_CR2	SCH_CR3	SCH_CR4	SCH_CR5	SCH_J8	SC
DESIGN	@hier_design_lib.processor(tbl_1):\I33\@hier_design_lib.analog_io(sch_1)									
1	12v									
2	afd									
3	afs									
4	agnd			b						
5										
6										
7	audout									
8	cclock									
9	data<3..0									
10	micin								con_pin<0>	
11	mtorf									
12	n12v							cat		
13		b								
14	rftum									
15	unnamed									
16	unnamed				an					

You can adjust the columns by dragging their borders for better display.

18. Close the report by selecting *File - Close*.
19. Click *No* to ignore the changes.

Summary

In this lesson, you learned to generate a cross tab report.

For More Information

See:

Creating Reports chapter of *Allegro Design Editor User Guide*.

Lesson 7-6: Creating Custom Columns in Reports

Overview

In this lesson, you will learn to create or edit user defined query fields that can be used in report templates.

Concept

A custom column is a user defined query field that can be used in report templates. For example, you can create columns with sub-columns.

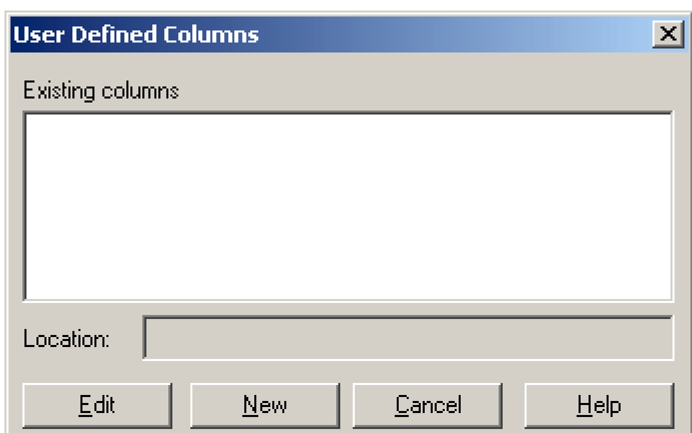
In this lesson, you will learn to create a column `COMPONENT INFORMATION`, which has two sub-columns—`PIN NAME` and `REFDES`.

Procedure

Creating a Custom Column

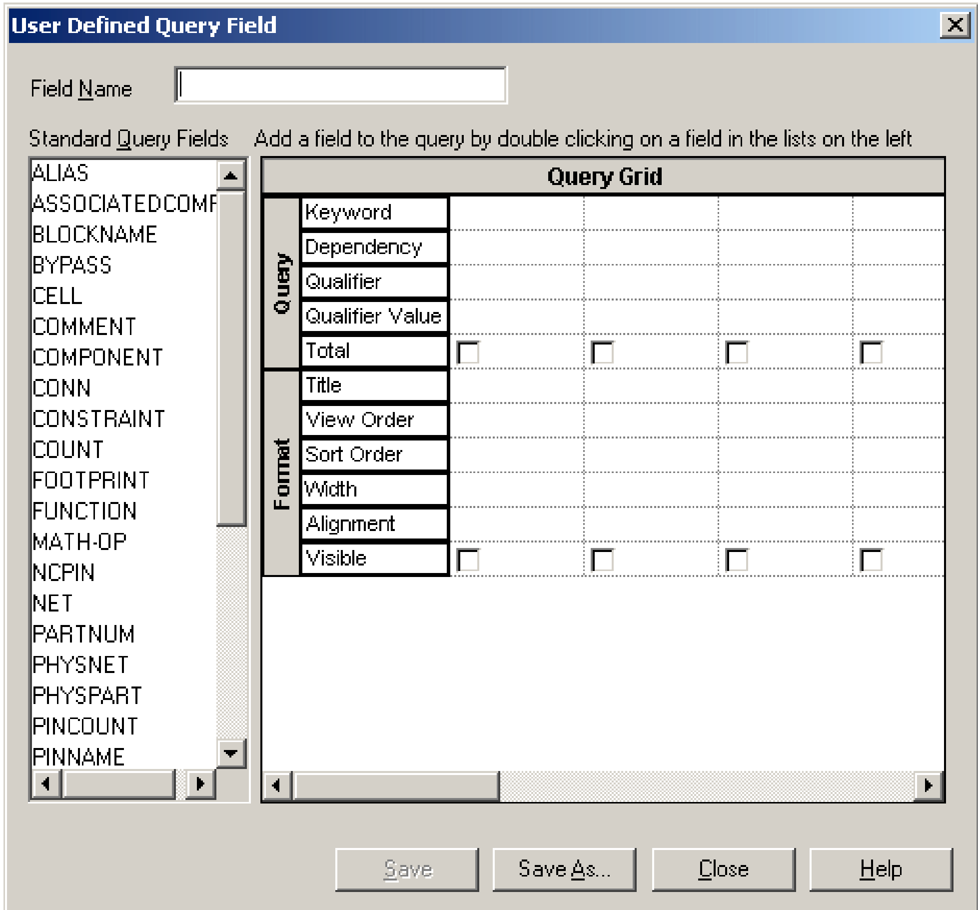
1. Choose *Projects - Reports - Create Custom Column*.

   The User Defined Columns dialog box appears.



2. Click *New*.

The User Defined Query Field dialog box appears.



Notice that the User Defined Query Field dialog box has a query grid similar to the query grid used in the Create Report Template dialog box

3. Type `COMPONENT_INFORMATION` as the field name.

4. Select the first keyword as `PINNAME`. To select the keyword, click in the first cell next to the *Keyword* cell and choose `PINNAME` from the list box.

5. Set the *Dependency* field for the `PINNAME` column to `PHYSNET`.

   This dependency will display the names of pins connected to the nets in the design in the report

6. Select the second keyword as `REFDES`.

Notice that the dependency is automatically set to `PINNAME` signifying that the report will display the reference designator of the components whose pins are connected to nets in the design.

7. Click the *Save As* button and save the report with the name `ComponentInfo.txt`.
8. Click the *Close* button to close the User Defined Query Field dialog box.
9. Click the *Cancel* button to stop making any more changes to the the user-defined columns.

Associating a Custom Column in a Report Template

You will now add the custom column in a report template and generate a report.

1. Choose *Projects - Reports - Create Template*.
2. Type `Connectivity Report` to define the *Report Name*.
3. Select the *Design Based* option button.
4. Select the first keyword as `PHYSNET`.
5. Select the second keyword as `COMPONENT_INFORMATION` by dragging the `COMPONENT_INFORMATION` keyword from the *User Defined Fields* to the cell next to the `PHYSNET` keyword.

   Notice that the title `COMPONENT_INFORMATION` appears in the `COMPONENT_INFO` column.
6. Click the *Save As* button and save the report template with the name `Connectivity_report.tpt`.
7. Click the *Close* button.

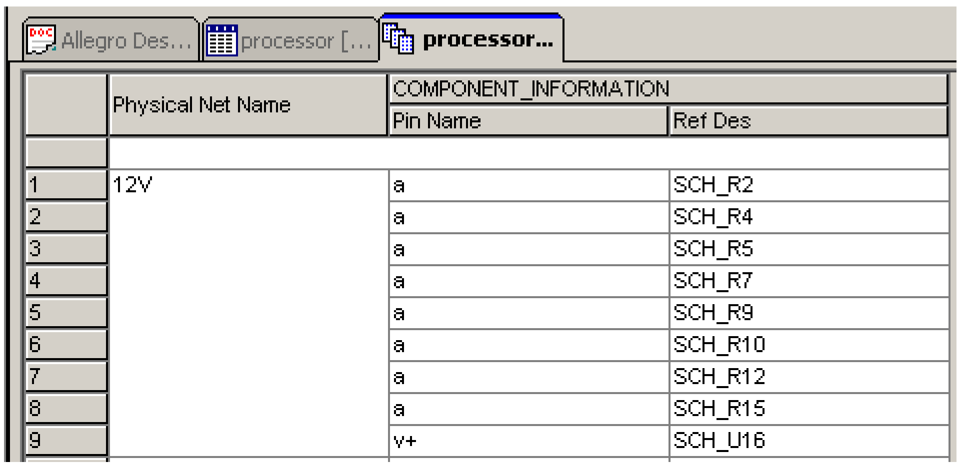
Generating a Report Containing Custom Columns

1. Choose *Projects - Reports - Generate Reports*.
2. Select *Connectivity Report* in the *Report Template* field.
3. Select the *DSR(Editable Report File)* option button to specify the report format.

4. Specify the output folder where the report will be located in the *Location* field.

5. Click the *Generate* button to generate the report.

   Notice that the generated report has a column titled `COMPONENT INFORMATION`, which has two sub-columns—`Pin Name` and `Ref Des`.



	Physical Net Name	COMPONENT_INFORMATION	
		Pin Name	Ref Des
1	12V	a	SCH_R2
2		a	SCH_R4
3		a	SCH_R5
4		a	SCH_R7
5		a	SCH_R9
6		a	SCH_R10
7		a	SCH_R12
8		a	SCH_R15
9		v+	SCH_U16

Summary

In this lesson, you learned to create custom columns, associate them with a report template and then generate reports.

For More Information

See:

Creating Reports chapter of *Allegro Design Editor User Guide*.

References

This appendix discusses the following:

- Learning More About Allegro Design Editor
- List of Sample Design Files
- List of Multimedia Demonstrations

Learning More About Allegro Design Editor

SourceLink®

SourceLink® online customer support gives you answers to your technical questions. Find the latest in quarterly software rollups (QSRs), case and product change release (PCR) information, technical documentation, FAQs, solutions, software updates, and more.

Important

To register on SourceLink you will need your email address and your host-ID or serial number.

To access SourceLink, go to:

http://sourcelink.cadence.com/

International Cadence Usergroup

The ICU is an independent, non-profit organization of users of Cadence products who meet annually at a conference in the fall and interact throughout the year, sharing solutions and ideas.

The ICU is independent of Cadence, but maintains strong ties at all levels in the company to conduct its business. Cadence representatives are invited to ICU board meetings and Cadence provides valuable assistance at the annual ICU conference (tutorials, executives, application engineers, and so on.). E-mail forums are also available where you can ask questions and seek information from other users.

For additional information, go to:

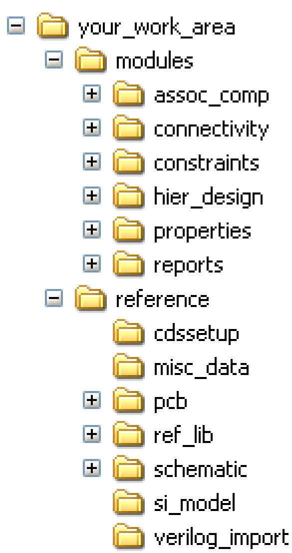
http://www.cadenceusers.org

List of Sample Design Files

The zipped or tarred copy of the tutorial database is located in the directory:

`<your_inst_dir>/doc/ade_tut/tutorial_examples/`

After you have unzipped the `ade_tut_db.zip` file or untarred the `ade_tut_db.t.Z` file, you will get the following directory structure.



The `modules` directory contains the design projects for each module in the tutorial. If you have not completed all the lessons in a previous module, you can open the project (`.cpm`) file for the current module and work through the lessons in the module.

The `reference` directory contains the part and footprint libraries and other files used in the tutorial. The important folders in the `reference` directory are described below:

Folder	Description
pcb	Contains the Allegro PCB footprint libraries.
ref_lib	Contains the part libraries and part table file used in designs in Design Editor.
schematic	Contains the schematic used in Lesson 6-2: Adding a Schematic Block in a Design of Module 6: Creating a Hierarchical Design.
si_model	Contains the signal integrity model library used in Lesson 5-3: Assigning Signal Integrity Models of Module 5: Working with Electrical Constraints.

List of Multimedia Demonstrations

The following table lists the multimedia demonstration files that accompany this tutorial. These demonstrations are installed in the following directory:

```
<your_inst_dir>/doc/ade_tut/tutorial_examples/demos
```

Click the Demonstration File Name in the table to launch a particular multimedia demonstration.

Note: Not all lessons have accompanying multimedia demonstrations. Some lessons do not require visual demonstrations.

This demo ...	shows how to ...	Time in Minutes
Creating a Project in Allegro Design Editor	Create projects in Design Editor.	8
Working with Components and Connectivity	Add components and signals in the design and capture connectivity information.	30
Working with Associated Components	Work with terminations, bypass capacitors and pullup/pulldowns in the design.	11

This demo ...	shows how to ...	Time in Minutes
Assigning Signal Integrity Models in Allegro Design Editor	Assign a signal integrity model on a component and automatically generate models for the two pin discrete components (resistors, capacitors and inductors) in the design.	7
Working with Hierarchical Designs	Work with hierarchical designs in Design Editor.	50
Generating Reports Using Allegro Design Editor	Create report templates and generate reports in Design Editor.	45