ViewDraw v6.3 User's Guide

Actel®

Actel Corporation, Mountain View, CA 94043

Printed in the United States of America

Part Number: 5-02-00025-1/10.05

Release: October 2005

Actel makes no warranties with respect to this documentation and disclaims any implied warranties of merchantability or fitness for a particular purpose. Information in this document is subject to change without notice. Actel assumes no responsibility for any errors that may appear in this document.

This document contains confidential proprietary information that is not to be disclosed to any unauthorized person without prior written consent of Actel Corporation.

Trademarks

Actel and the Actel logotype are registered trademarks of Actel Corporation.

Adobe and Acrobat Reader are registered trademarks of Adobe Systems, Inc.

Mentor Graphics, Precision RTL, Exemplar Spectrum, and LeonardoSpectrum are registered trademarks of Mentor Graphics, Inc.

WaveFormer Lite is a registered trademark of SynaptiCAD, Inc.

Synplify is a registered trademark of Synplicity, Inc.

Sun and Sun Workstation, SunOS, and Solaris are trademarks or registered trademarks of Sun Microsystems, Inc

Synopsys is a registered trademark of Synopsys, Inc.

Verilog is a registered trademark of Open Verilog International.

Viewlogic, ViewSim, ViewDraw and SpeedWave are trademarks or registered trademarks of Viewlogic Systems, Inc.

Windows is a registered trademark and Windows NT is a trademark of
Microsoft Corporation in the U.S. and other countries.

UNIX is a registered trademark of X/Open Company Limited.

All other products or brand names mentioned are trademarks or registered trademarks of their respective holders.

Table of Contents

Getting Started With ViewDraw

ViewDraw for Actel

ViewDraw is a graphical schematic design tool that helps you create symbol and schematic designs.

ViewDraw is a Windows-based application that uses standard Windows user interface controls. You create designs using menu commands, toolbar buttons, and by selecting and entering information on dialog boxes.

ViewDraw AE is a special version of ViewDraw. Use it for schematic entry with Libero® Integrated Design Environment (IDE).

Co-installation of ePD or DxViewDraw from Mentor Graphics with ViewDraw AE in Libero IDE is supported in Libero IDE v6.0. You can switch between ePD ViewDraw and ViewDraw AE by running *configurator* of both tools.

Note: To run ViewDraw, you must log in as a Power user or Administrator. If you log in as a Normal user, ViewDraw generates an error when it is invoked.

See Also

Switching Configuration Between ViewDraw Installations (co-installed)

Schematic Guidelines

When creating your schematics, follow these guidelines.

Using Hierarchical Connectors

Hierarchical connectors are required in any schematic design for external ports for the top level. Hierarchical connectors must also be used on ports of sub-modules.

I/O Pads

I/O pads can be automatically inserted by running synthesis. For schematic designs that only use Actel primitives (no HDL blocks), you can manually insert ALL I/O pads if you do not want to go through the synthesis flow.

Naming Conventions

Project names must be less than 8 characters.

"my_top" is acceptable

"my_top_level" is not

Do not use spaces in:

- Project names
- Instance names
- Net names
- Port names
- Schematic names
- Stimulus file names
- HDL names, etc.

Do not use any special characters in any of your naming, such as: ~ ! @ # $ % ^ & * ( ) = + { } | \ / < > ? ` ' " ", . or spaces.

- The "inverted" net property is not supported.
- If you want to rip out scalar bits from a bus, use [] for scalar bit naming. For example, Bus[15], Bus[14], …, Bus[0].
- Do not use numbers at the beginning or the end of any names. ViewDraw AE regards Bus[1] as equivalent to Bus1.
- Scalar bit Bus1[15] of Bus1[15:0] conflicts with scalar bit Bus11[5] of Bus11[15:0] during netlist generation.
- If you want to use numbers to distinguish related nets, numbers can be used followed by letters at the end, for example: NET1N, or Bus1A[15:0].
- Multi-dimensional busses are not supported. For example, do not use naming "Bus[0:3][0:3]" in ViewDraw AE.
- Do not name any component same as Actel library components.

Finding an Inverted Signal in a Schematic

When looking for an inverted signal, use '~signal_name' instead of 'signal_name', since in schematic and wir files, inverted signals are saved as '~signal_name'. Use **Edit – Select** option.

Function Key Guide and Hotkeys

F1	F2	F3	F4	F5	F6	F7	F8	F9	F10
Help	Left Mouse	Right Mouse	View Full	Refresh	Pan	View In	View Out	View Zoom	

Click here for a list of PC Key Bindings.

The viewdraw.ini File

Libero IDE creates an initialization file in the ViewDraw sub-directory created under project directory. The file name is viewdraw.ini. This file is a text file that specifies the operating configuration of the ViewDraw application. ViewDraw reads this initialization file every time you run ViewDraw.

The viewdraw.ini file contains keywords followed by parameters. The keyword identifies a particular setting (such as the default color for a net). The parameters are default settings set by Libero IDE. Lines in the viewdraw.ini file that begin with vertical bars (|) are considered comments and are ignored.

You can modify the viewdraw.ini file when you change the settings in the Project Settings dialog box. You can also edit the file manually using any text editor.

You can use a viewdraw.ini file that contains search paths for the PC.

Each time you invoke ViewDraw from Libero IDE, the tool uses the viewdraw.ini file in the viewdraw directory in the project created by Libero IDE.

See Also

Sample viewdraw.ini File

Switching Configuration Between ViewDraw Installations (co-installed)

To invoke ViewDraw immediately after installation:

If you want to use ViewDraw immediately after installing it, just invoke it through Libero IDE or ePD.

Using ViewDraw A.E. (Libero IDE v6.0) after running a different co-installed ViewDraw

To reset the registry and environment variables to support ViewDraw Actel Edition from Libero IDE v6.0, you must:

- Close any ViewDraw windows. You may need to reboot a machine to ensure that all background services for ePD or ViewDraw are stopped.
- Go to the ViewDraw folder in Libero IDE v6.0 installation, <INSTALLATION>/ViewDraw/, and run configurator.exe.
- Double-check that the PATH variable contains the location of ViewDraw A.E. for Libero IDE v6.0 as the first entry.
- Invoke Libero IDE v6.0 and startup ViewDraw.

Using ViewDraw A.E. (Libero IDE v5.x) after running a different co-install ViewDraw

To reset the registry and environment variables to support ViewDraw Actel Edition from Libero IDE v5.x, you must:

- Close any ViewDraw windows. You may need to reboot a machine to ensure that all background services for ePD or ViewDraw are stopped.
- Go to the ViewDraw folder in Libero IDE v5.x installation, <INSTALLATION>/ViewDraw/, and run configurator.exe.
- Double-check that the PATH variable contains the location of ViewDraw A.E. for Libero IDE v5.x as the first entry.
- Invoke Libero IDE v5.x and startup ViewDraw.

Using an earlier ViewDraw A.E. (Libero IDE v1.0 to v2.3) after running a different co-install ViewDraw

To reset the registry and environment variables to support ViewDraw Actel Edition from Libero IDE v2.x, you must:

- Download a copy of the Libero IDE v5.0's ViewDraw A.E. configurator.exe and put this into the <INSTALLATION>/ViewDraw/ folder for the version of ViewDraw A.E. that you want to use.
- Close any ViewDraw windows. You may need to reboot a machine to ensure that all background services for ePD or ViewDraw are stopped.
- Go to the ViewDraw folder in Libero IDE installation that you just copied the configurator.exe into, and run configurator.exe.
- Double-check that the PATH variable contains the location of ViewDraw A.E. for Libero IDE v2.x as the first entry.
- Invoke Libero IDE and startup ViewDraw.

Using eProduct Designer (2.0, 3.0) after running ViewDraw Actel Edition

To reset the registry and environment variables to support ViewDraw from ePD, you must:

- Close any ViewDraw windows. You may need to reboot a machine to ensure that all background services for ePD or ViewDraw are stopped.
- From Start > Programs, go to eProduct Designer and run Configure eProduct Designer.
- Startup ePD.

Known Problems and Workarounds

eProduct Designer Configurator Warning

When running the ePD configurator after you have installed Libero IDE v6.0, you may get a warning about 'PATH not set correctly'. You can ignore this warning. Both eProduct and Libero IDE will run properly.

Getting Help

You can access the following types of help:

Context Sensitive Help

Context sensitive help is built into every menu command, toolbar button, and dialog box. Just position the cursor over the item you want help on and press F1. Context sensitive help is also available when you click ? and then click on a menu command or toolbar button. For dialog box help, click the Help button on the dialog box.

Instructional Help

Instructional help is available from the ViewDraw Help menu. This menu offers ways to get help on using the Help facility, including instructions on searching for help topics, and help on using ViewDraw.

Reference Material

Information such as commands, toolbar buttons, associated utilities, and specific syntax descriptions is also available from the ViewDraw Help menu.

Troubleshooting Help

Error messages and required user actions are available when you press **F1** on an active message box. This information is also available from the ViewDraw Help menu.

Understanding the Hierarchy

ViewDraw uses block types to indicate hierarchical levels in a design. A composite block indicates a hierarchical level that underlying blocks (at the next level down the hierarchical tree) define. A module block indicates (1) a primitive symbol that has no hierarchy (the bottom of the tree), or (2) a symbol whose behavior is described by an external model file. Design primitives (such as the Built-in symbols or VHDL models) represent the lowest level in the hierarchy. ViewDraw supports an unlimited number of hierarchical levels.

A simple hierarchy example is as follows:

- Level 1 – Top level schematic (which includes a 74LS148 symbol)
- Component Layer -- A symbol for a component on the top level schematic that references the lower level of the hierarchy.
- Level 2 – The underlying schematic for the symbol

The symbol and its underlying schematic make up one level of the hierarchy.

For example:

The top-level schematic represents the highest level of the schematic. The component 74LS148 is one of many components on the schematic.

The symbol of 74LS148 represents the connectivity to the next level of the hierarchy. This symbol is a composite block that has an underlying internal schematic.

The underlying internal schematic for the symbol 74LS148 is the lowest level of the hierarchy. The schematic 74LS148 is made up of Built-in symbols. Built-in symbols represent a design primitive at the lowest level of the hierarchy.

You can move down through your design using the Schematic and Symbol pop up or the windows menu.

When you are ready to simulate the design, a wirelist program interprets the design levels for input to the simulator. A composite type component indicates to the wirelist program that it should trace down the block hierarchy. When a wirelist encounters a module block, it stops at that level in the hierarchy.

Moving Through the Levels of the Hierarchy

You can explore the hierarchy of a schematic using the following options:

To page through the sheets of a schematic use the Page Up and Page Down keys or use the GoTo, Previous Page, or Next Page options from the pop up menu.

To move to a different level or sheet in the schematic or symbol:

1. Right-click a component in the active window to get the shortcut menu.

   The title bar of the active window indicates the schematic or symbol's name.

2. Use the Schematic or Symbol options from the shortcut menu.

These menus allow you to go to any sheet in your design. To return up in the design, either close the lower layer windows or use the Windows menu to select the desired window.

Understanding Blocks and Sheets

Blocks and sheets help you better separate your work. Blocks help you group your design into common types. A block is a schematic or a symbol sheet.

The symbol block border is a frame for creating the symbol. When you add a pin to the symbol body, the pin snaps to the nearest edge of the border. ViewDraw does not display the symbol border when a component is placed on the schematic.

The default symbol block size is 1" by 1". The system defines this as the Z-size default setting. You can change the block size setting using the Properties dialog box.

ViewDraw uses the following block types in the design hierarchy:

- Composite
- Module
- Pin
- Annotate

Composite Block

You define a composite block with a corresponding internal schematic or wirelist. Use composite blocks to signify hierarchy within the design. The underlying schematic is another schematic of the hierarchy. A composite type component indicates to the wirelist program that it should trace this component down the design hierarchy.

Any instance of a composite component on a schematic must have an underlying schematic.

The LEVEL attribute specifies a control of level for wirelisting. You can specify a level when netlisting to determine the lowest level to be used. The value of the LEVEL attribute for the lowest level is HARD.

Module Block

A module block is a symbol with no corresponding internal schematic. A module block exists at the lowest level of the hierarchy and does not require a definition. Another name for a module block is primitive. Use module blocks to create flat designs.

Module blocks indicate to the wirelist program that the component is a primitive that has no hierarchy. When a wirelist encounters a module block, it stops at that level in the hierarchy. Because module blocks have no underlying schematic, they do not have WIR files. You describe a module function by a primitive or some external file.

Example

VHDL and ABEL models are examples of modules. These symbols do not have underlying schematics. You describe their function with an external model (.vhd, .abl) file.

Pin Block

Use a pin to represent a schematic pin. A component of a pin symbol specifies for the wirelist that the net connected to this component is an interface (port) signal and represents a connection to a specific pin on the symbol of the corresponding schematic. Labeling the component with the same name as the symbol pin specifies a connection down through the hierarchy.

Pin blocks also determine net names without labels. You can tie a net to a global signal (such as GND, VDD, RESET, etc.). A pin symbol that represents a global signal must have the NETNAME attribute.

You add the NETNAME attribute to pin symbols to automatically label nets. For example, specify the NETNAME = GND in the attribute field of the Pin Properties dialog box and ViewDraw automatically labels the net connecting to this component as GND.

If the signal used with the NETNAME attribute is a global signal, it should be declared in the viewdraw.ini file as such. You can add new globals using the Project Settings dialog box.

Annotate Block

An annotate block is a symbol with no electrical definition (that is, isolated from schematic connectivity). Use an annotation block to document a circuit.

Wirelisting ignores annotate blocks when placed as components in a schematic. This allows you to place documentation graphics in a schematic (such as title boxes, borders, etc.).

Determining the Block Type

The **Memory** command on the Project menu displays a list of all block in active memory.

To list all block types:

1. Choose Project – Memory.
2. View the active output window.
3. Depress the right mouse button and select Save As to save the information.

If you do not want to save the information in the output window, close the window without saving.

Block Type Codes:

- C Composite
- M Module
- P Pin
- A Annotate

Changing the Block Type of New Symbols

You can change the block type for new symbols by using the Project Settings dialog box.

To change the block type for new symbols:

1. Choose Project – Settings.
2. Select the Block tab on the Project Settings dialog box.
3. Change the symbol type option to the correct block type.

4. Click **OK** to save the settings and dismiss the Project Settings dialog box.

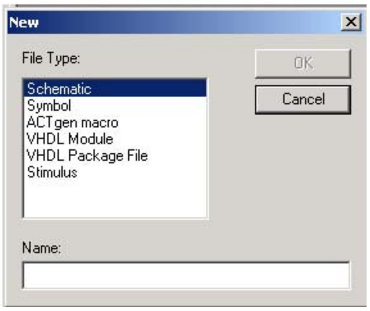
To change the block type for existing symbols:

1. Open the symbol.
2. Double-click a blank area of the symbol to get the Symbols Properties dialog box.
3. Select the desired block type.
4. Click **OK** to accept the change and dismiss the dialog box.

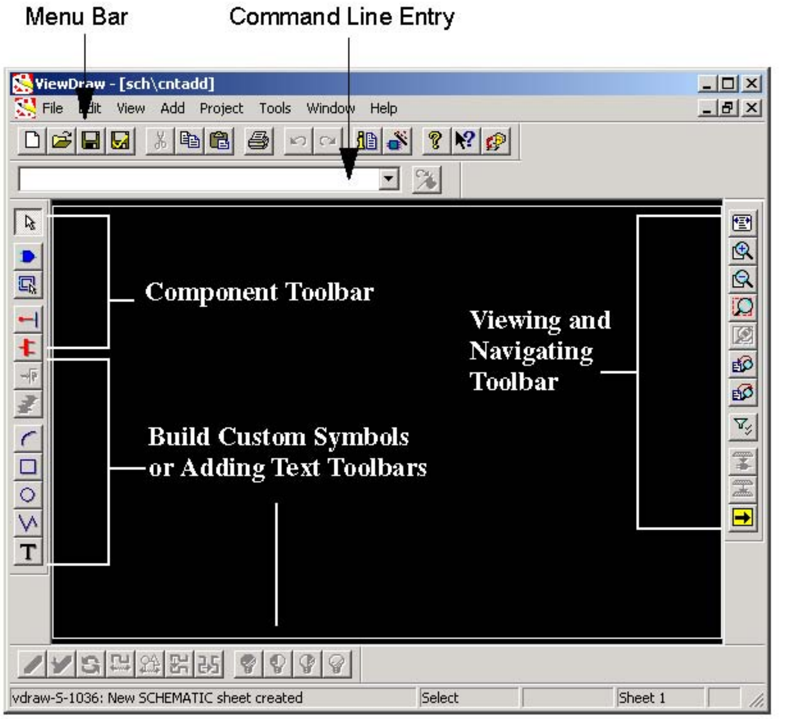
Using ViewDraw With Libero IDE

Starting ViewDraw

1. Open your project in Libero IDE.
2. From the File menu, click New. This displays the New dialog box, as shown below.



3. Click Schematic and type a name for your schematic file in the Name field. Click OK. ViewDraw for Actel starts, as shown below.



vdraw-S-1036: New SCHEMATIC sheet created

Figure 5-3. ViewDraw for Actel

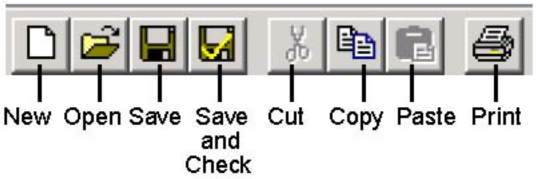
Now you are ready to create a new schematic by adding components that represent design primitives or by placing components of symbols that you have created or modified on to the schematic sheet. Once you have placed the components add nets and busses to form connections between the components, (see "Adding Nets and Buses for Connectivity" for more information.)

Toolbars

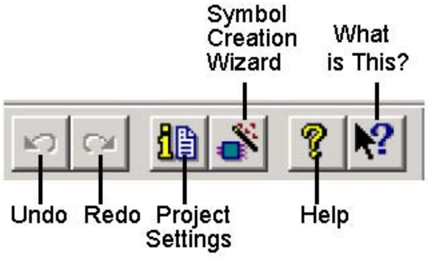
ViewDraw for Actel contains 4 toolbars: Standard, Object, Transform, and View.

Standard Toolbar

Use the Standard toolbar for your basic project functions.



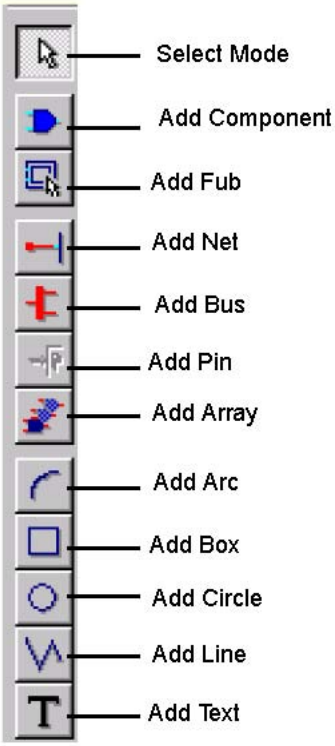
ViewDraw Standard Toolbar



ViewDraw Standard Toolbar, continued.

Object Toolbar

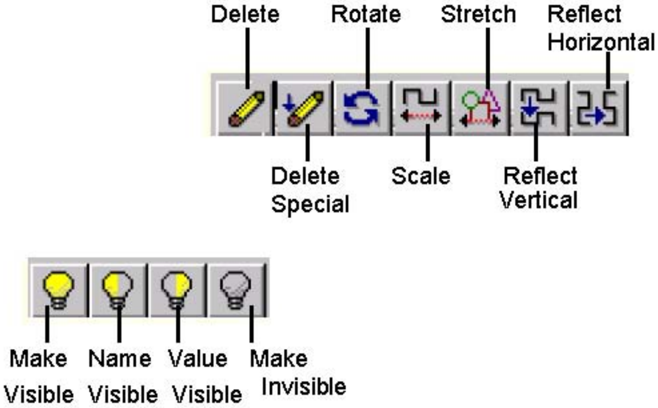
Use the object toolbar to add components, create symbols, and add text.



Object Toolbar

Transform Toolbar

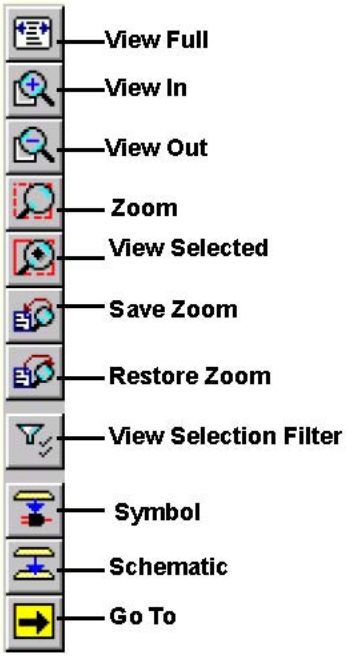
Use the transform toolbar to create custom symbols.



Transform Toolbar, continued

View Toolbar

Use the View toolbar to navigate and to change your view.



View Toolbar

Creating a Schematic Source File

Use ViewDraw AE to create your schematic source files.

To create a schematic source file:

1. From the **File** menu, click **New**.
2. Select **Schematic** and type a name for your schematic file in the Name field. Click **OK**. ViewDraw AE starts.
3. Using ViewDraw AE, create your schematic.
4. When you are done, click **Save + Check**. The **Save + Check** command creates your WIR file. When Save and Check is complete, the message "Check complete, 0 errors and 0 warnings in project <name>" appears in the status bar. You must select Save & Check. Only selecting Save will not generate the needed WIR file for that block.
5. (Optional) Run connectivity checker. Right-click the schematic file in the File Manager tab and select **Check Schematic**. The connectivity checker checks the connectivity of the wir file. Errors and warnings appear in the log window.
6. From the **File** menu, click **Exit**. The schematic is saved to your project in Libero IDE, appearing in both the File Manager and the Design Hierarchy tabs.

Importing Schematics

You can import any schematic created with ViewDraw AE using Libero IDE Import Files option.

To import a schematic file:

1. From the Libero IDE **File** menu, click **Import Files**.
2. In Files of type, select Schematics.
3. In Look in, navigate to the drive/folder where the file is located.
4. Select the file to import and click **Open**. The schematic is imported into your project and appears in the File Manager, under Schematic files as well as Design Hierarchy window.

To open the schematic, highlight the schematic file in Design Hierarchy or File Manager window and double-click or right-click the file in the File Manager and select, **Open Schematic**.

Note: Only importing schematic file does not import wir file. You need to click Save + Check for each schematic block to generate wir files. Otherwise, the flow cannot continue since the HDL files are generated from the wir files.

Schematic File Open

Use ViewDraw AE to edit your schematic files.

To open your schematic file:

1. **Open** your project in Libero IDE.
2. Double-click the schematic file in the File Manager or Design Hierarchy windows. ViewDraw AE opens with the file loaded.
3. From the **File** menu, click **Save + Check** to create the required files for netlist generation. When **Save + Check** is complete, the Status Bar will say "Check complete, 0 errors and 0 warnings in project <name>." You must select **Save + Check**. Only selecting Save will not generate the needed WIR file for that block.
4. From the **File** menu, click **Exit**. The schematic is saved to the project, appearing in both the File Manager and Design Hierarchy tabs. Your schematic file is updated in Libero IDE.

Using ACTGen Cores

Use ACTgen to:

- create high level modules, such as counters, multiplexors, multipliers, etc. that are optimized for Actel FPGAs.
- create system level building blocks, such as filters, FIFOs and memories.

These can be instantiated into your schematic, Verilog design, or HDL design.

To generate cores for your schematic:

1. Add the ACTgen core to your Libero IDE project.

- From the Libero IDE **File** menu, click **New**.
- In the **New File** dialog box, select ACTgen core, type a name, and click **OK**. ACTgen starts.
- Select your core type from the left core list box. The appropriate options appear. Select a tab and fill in the fields. Click Generate to create an HDL representation of the core.
- In the **Save As** dialog box, leave the default selections and click **Save**. The file is added to your Libero IDE project, appearing in the Design Hierarchy.

2. Create the Symbol. In the Design Hierarchy, right-click the ACTgen module and choose **Create Symbol**. The symbol is created, appearing in the File Manager, under Block Symbol files.
3. Use the Symbol.

- Start ViewDraw.
- From the Add menu, click Component.
- Select the new symbol, then drag and drop it onto your schematic.

Customizing the Tool Menu

Customizing the Tools Menu

You can customize the Tools menu to include programs that you want to launch from your ViewDraw application.

Customize your Tools menu by adding, removing, or editing menu command entries.

The Tools menu has three sections:

System Menu Commands

Commands that the application places on the Tools menu. These commands appear in the top most section of the Tools menu command list. You cannot customize or edit these commands.

Common Menu Commands

Commands that you place on the Tools menu that are available to any user who logs on to the PC. These commands appear in the middle section of the Tools menu command list.

Viewdraw stores changes to the common menu in a file named commontools.ini that is located in the Libero IDE/ViewDraw/standard directory. These changes will be effective for all users of this machine in all projects.

User Menu Commands

Commands that you place on the Tools menu that are associated with your logon information and available only when you log on to the PC. These commands appear in the bottom most section of the Tools menu command list.

ViewDraw stores information about the user menu in a file named usertools.ini. ViewDraw maintains copies of this file in each project and in the first writable directory in your WDIR path. ViewDraw will use either the copy of usertools.ini in the current project or the copy in the WDIR directory. You control which file ViewDraw uses through the Customize Tools Menu Dialog Box.

Adding a Command to the Tools Menu

To add a command to the Tools menu:

1. Choose **Customize** from the Tools menu to display the **Customize Tools Menu** dialog box.
2. If you want the command available to all users who log on to your PC, choose **Common** menu from the **Menu** section. **User Specific** menu is the default.
3. Enter the name that you want to appear as the new menu command in the **Menu Text** field. Specify a letter in the menu title as a menu accelerator by entering the title with an ampersand (&) immediately preceding the accelerator letter. If you do not specify an accelerator, the first unique letter in the title is the accelerator by default.
4. Enter the command that invokes the new application in the **Command** field.

   Use the **Browse** button to select the appropriate drive and directory, and then select the executable you want to add from the list of file names.
5. Enter the arguments associated with the command in the **Arguments** field. For information on valid arguments, refer to Using Argument.
6. Enter the working directory for the tool in the Initial directory field.
7. Click the **Add** button to add the menu item to the Tools menu.
8. Click the **OK** button to dismiss the dialog box.

The command now appears on the Tools menu. To run the program, choose it from the menu.

Editing a Tools Menu Command Entry

To edit a Tools menu entry:

1. Choose **Customize** from the Tools menu to display the Customize Tools Menu dialog box.
2. Click the menu option that you want to edit from the **Menu Contents** field. When you select the option, the information associated with that option appears in the Menu Text, Command, Argument, and Initial Directory fields.
3. Edit the field that you want to change. For example, edit the command name that appears on the Tools menu by editing the text in the Menu Text field. You can also change the location of the menu item in list using the Move Up and Move Down buttons.
4. Click **OK** to accept the changes and dismiss the Customize Tools Menu dialog box.

   The application updates the Tools menu to reflect your changes.

Removing a Command From the Tools Menu

To remove a command from the Tools menu:

1. Choose **Customize** from the Tools menu.

   The Customize Tools Menu dialog box appears.
2. In the Menu Contents field, click on the command you want to remove.
3. Click the **Remove** button to remove the command from the Tools menu.
4. Click **OK** to dismiss the Customize Tools Menu dialog box.

   The application updates the Tools menu to reflect your changes.

Using Arguments

You can specify arguments for any program that you add to the Tools menu. Each application supports a set of predefined variables called arguments. Arguments are not required.

Enter arguments (in uppercase) in the Arguments field of the Customize Tools Menu dialog box. If you want to use more than one argument, leave a space between each argument entry.

Note: If the command is named the same as the executable, the application closes the window when done executing. If you want to keep the window open, use the /k qualifier as the first argument.

ViewDraw Arguments

Argument entries are case sensitive and must be entered in uppercase.

Argument	Description
$BLOCKNAME	The file name of the current symbol or schematic.
$BLOCKPAGE	The current sheet of the schematic.
$BLOCKTYPE	A string that defines the type of drawing (SCHEMATIC or SYMBOL). This string is always uppercase.
$PROJDIR	The path to the current project directory.
$COMPNAME	The component label of the selected component.
$NETNAME	The net label of the selected net.
$PINNAME	The pin label of the selected pin.

Synthesis Arguments

Argument entries are case sensitive and must be entered in uppercase.

Argument	Description
$ENTITYNAME	The selected entity in the entity tree.
$PROJDIR	The path to the current project.
$PROJNAME	The name of the current project.

Customize Tools Menu dialog box

Adds, edits, or removes Tools menu commands.

Dialog Box Options

Menu Item Types

Common

Specifies that the command is common to users who log on to the PC. If you select this option, the command appears on the Tools menu every time the application runs on that PC, regardless of which user has logged on.

User Specific

Specifies that the command is specific to your logon information. If you select this option, the command appears only when you log on to the PC.

Customize this project only

This check box is only activated when the User Specific button has been checked It specifies where the commands you add will be available.

If you check this box, the commands that you add will only be available to the current project.

If you clear this box, the commands that you add will be available to all projects. The box is cleared by default.

Menu Contents

Lists the commands that you have added to the Tools menu. To edit a command entry, select the command from this list.

Menu Text

The command name you specify as you want it to appear on the Tools menu.

You enter the menu text in this field. If you want to assign a keyboard accelerator to the command, enter an ampersand (&) immediately before the letter you want used as the accelerator. If you do not specify an accelerator, the first unique letter of the command name is used.

Command

The command associated with the command name that executes the program you add.

Arguments

Each application provides a different set of predefined variables that you can use. For a list of available arguments, refer to Using Arguments.

Initial Directory

The working directory for the tool. If you do not specify a directory, the system defaults to the current directory.

Dialog Box Buttons

OK

Accepts changes if required and dismisses the dialog box.

Cancel

Ignores changes and dismisses the dialog box.

Add

Adds the new command to the Tools menu.

Remove

Removes the command selected in the Menu Contents section from the Tools menu.

Move Up

Moves the command selected in the Menu Contents section up one position in the menu list.

Move Down

Moves the command selected in the Menu Contents section down one position in the menu list.

Browse

Browses through drives and directories to find the program you want to add to the Tools menu.

Help

Opens this Help window.

Creating Schematics

Creating a New Schematic

Create a new schematic by adding components that represent design primitives or by placing components of symbols that you have created or modified on to the schematic sheet. Once you have placed the components, then you can form connections between the components by adding nets and buses.

Each of the following topics is part of the process used to build a schematic:

- Creating a New Schematic
- Adding Components
- Adding a Fub
- Adding Nets and Buses
- Adding Labels
- Adding Attributes

Creating a New Schematic

Once you have established a project for your design, create a new schematic sheet to begin designing.

To create a new schematic

1. Choose File - New or click the 🗋 button.
2. Choose **Schematic** as the document type.
3. Enter a name and click **OK** to create the schematic.

See Also

Adding Components

Grounding Individual Nets of a Bus

Creating Multi-Page Schematic

To create multiple-page schematics

1. Create one page as normal and click **Save** + **Check**.
2. Click ➡ to go to a new blank page.
3. Create the schematic and click **Save** + **Check** again.
4. Repeat the above steps until you create all pages you want.

To switch between multi-page schematics

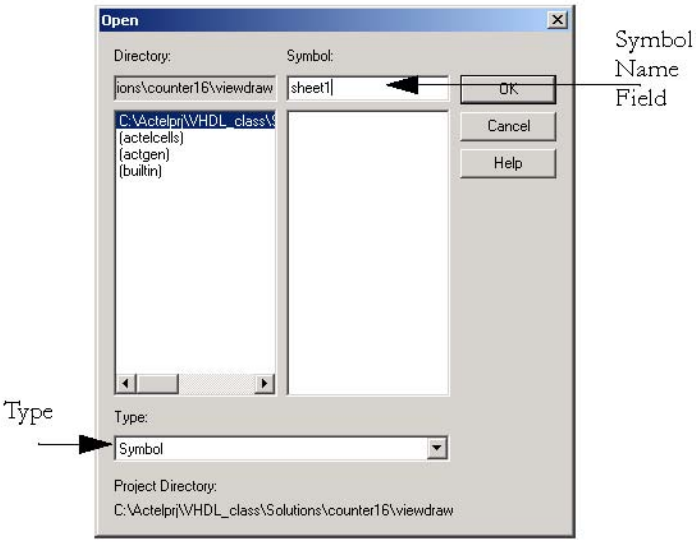
1. Open a page same as open one-page schematic
2. Click ➡ to go to the other page.
3. Specify the number of the page you want to go.

Creating a Sheet Border

You can create a customized sheet border for any size ViewDraw schematic. After you prepare the sheet border, it appears automatically on new schematics. The Builtin Library contains prebuilt sheet borders. If you want to use one of the prebuilt sheet borders, select the ASHEET, BSHEET, CSHEET, etc. symbol from the builtin library. Or, you can use the following instructions to create and add a custom sheet border.

To create a sheet border:

1. Choose File - Open to display the Open dialog box as shown below.
2. Choose **Symbol** as the Type.



Open

Directory: ions\counter16\viewdraw

Symbol: sheet1

C:\Actelprj\VHDL_class\S
(actelcells)
(actgen)
(builtin)

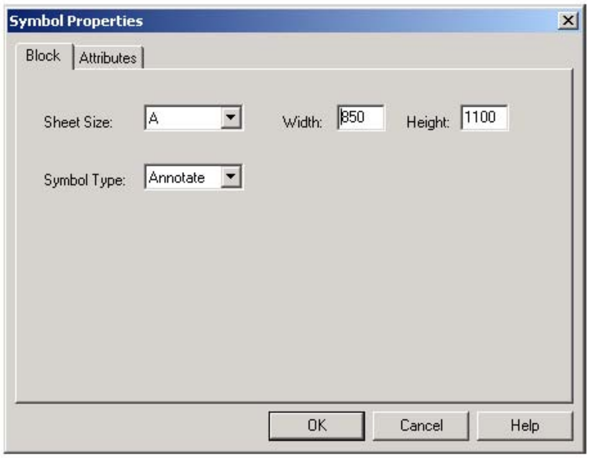
Type: Symbol

Project Directory:
C:\Actelprj\VHDL_class\Solutions\counter16\viewdraw

Open Dialog Box

3. Enter a name for the border in the Symbol field.
4. Click **OK** to open the symbol window.

5. Double-click the symbol window and change the block type and size.



Sheet Size: A
Width: 850
Height: 1100
Symbol Type: Annotate

Changing the Sheet Size Using the Symbol Properties Dialog Box

6. Change the sheet size to the size you want to have as your default sheet and change the block type to Annotate.
7. Click **OK** to accept the changes and dismiss the Symbol Properties dialog box.
8. Draw the border using the **Arc, Box, Circle**, and **Line** commands from the **Add** menu.
9. Double-click on the symbol window and change the block size to 1 x 1.
10. Choose File - Save to save the border.

The symbol can be saved to any location in your search order.

To add the border to a schematic:

1. Choose File - Open to display the **Open** dialog box.
2. Choose **Schematic** as the Type.
3. Enter a name for the schematic in the **Schematic** field.

This must be of the form <size>sheet.1. For example, the name would be asheet.1 or bsheet.1 for a standard size sheet or zsheet.1 for a custom sheet.

4. Choose Add - Component or click the button.
5. Click and drag the border component from the list or the symbol preview window to the lower left corner of the schematic.
6. Release the button to place the symbol.
7. Choose **Settings** from the **Project** menu and select the **Block** tab.
8. Check the **Add Default Sheet** option.
9. Change the sheet size option to the same size as your border component (For example, A, B, or Z).

If you choose Z, you must specify the sheet dimensions in ViewDraw units. These dimensions should match those which you specified as your block size when you drew the border graphic.

10. Choose File - Save to save the schematic.

ViewDraw saves the schematic to the sch subdirectory.

Once the sheet border schematic is created, the DEFSHEET and SHEETSIZE settings in the viewdraw.ini file are modified and all new schematics will have the sheet border. To add the border to existing components, use the Component option from the Add menu to add the component representing the sheet border symbol to the lower left corner of the schematic.

See Also

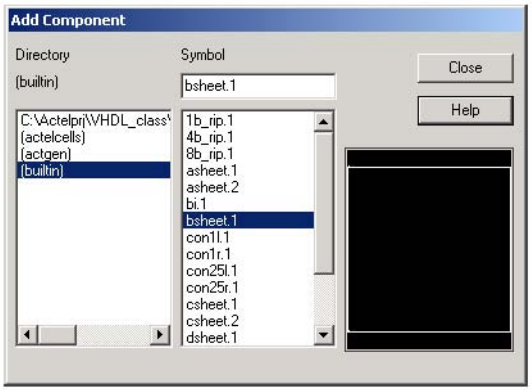
Adding Components

Creating a New Schematic

Selecting a Sheet Size

To change the sheet size:

1. From the **Add** menu, click **Component**. The Add Component dialog box appears, as shown in he figure below.



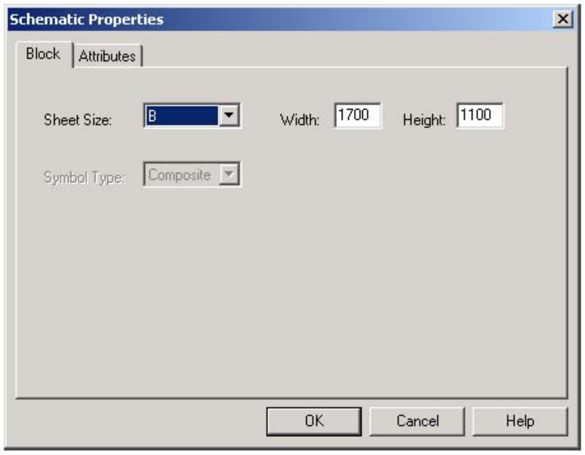
Selecting Sheet Size

2. Select the desired sheet size in Actel ViewDraw Builtin Library. (For example, asheet.1, bsheet.1)
3. Drag and drop the sheet into your schematic.

Changing the Sheet Size

To change the schematic sheet size in ViewDraw:

1. Right-click the schematic background.
2. Select Properties from the right-click menu. The Schematic Properties dialog box appears.



Sheet Size: B

Width: 1700

Height: 1100

Symbol Type: Composite

Schematic Properties Dialog Box

3. Choose your preference and click **OK**.

Showing Time and Date in the Schematic

To show the time and date in the schematic:

1. Right-click any blank space in the schematic, select **Properties** and click **Attributes** from the right-click menu.
2. Add attribute @DATETIME.
3. Click **OK**. The time and date will be update every time you modify and save the schematic.

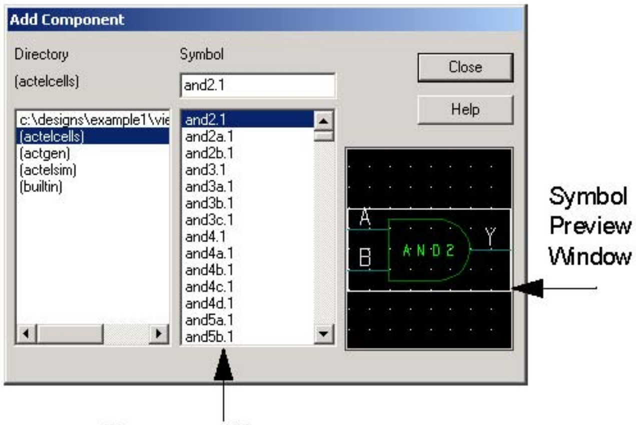
Adding Components

Components allow you to use a symbol many times in a schematic. You use the Component command from the Add menu or the button to add components to the schematic. The command-line command for this function is **comp**.

To add a component to the active schematic window

1. Choose Add - Component or click the button.

The **Add Component** dialog box appears.



Symbol Preview Window

Component Names

Add Component Dialog Box

2. Press and hold the mouse button on the component name or the symbol preview window to select the component you want to add from the list.

The symbol appears in the symbol preview window in the lower right corner of the dialog box.

3. Drag the component from either the component list or the symbol preview window onto the schematic sheet.
4. Release the button to anchor the component on the schematic.
5. Click **Cancel** to dismiss the **Add Component** dialog box.

Note: Pressing ESC while dragging the component cancels the operation.

You can keep the Add Component dialog box open as you edit the schematic. This allows you to add components at any time without reopening the Add Component dialog box for each component you add.

Adding a Fub

Click here for a description of fubs and how they work.

To add a functional block to your design:

1. Choose Add - Fub from the Add menu or click the button.
2. Click and drag to define the functional block area.
3. Release the mouse button.
4. Enter the Fub name in the **Add Fub** dialog box.
5. Click **OK** to add the Fub to the design.

The Fub appears as a symbol shell in the design.

6. Draw the required Nets and Buses connecting to and from the Fub.

ViewDraw automatically adds the pins at this time.

7. Label each Net and Bus.

After you save the design, ViewDraw automatically updates the symbol shell to include the required pin labels by copying the labels from the attached nets.

8. Select the Fub.
9. Press and hold the right mouse button to select the **Symbol** option; this pushes you into the symbol.
10. With your cursor outside the symbol box, right-click and hold to select the **Properties** option.
11. Choose the **Attributes** Tab.
12. Remove the SYMTYPE attribute from the Fub.
13. Add other necessary attributes to the Fub.
14. Choose **File - Save** or click the 💾 button.

Your Fub is now a symbol that you can add to designs using the Add Component command.

Note: Press Esc while dragging the box (or select Cancel in the dialog box) to cancel the Add Fub operation.

Connecting Components

Once you place components on a schematic, you can connect them using nets and buses. You construct a net or a bus under a specified routing mode that assists with determining the optimal route path.

The following types of connections are valid:

- Intersecting Connections
- Dangling Connections
- Connections by Abutment
- Connecting Identical Labels

See Also

Adding Components

Creating a New Schematic

Copying Components

You can duplicate a component or object to place a copy at a different location in the active window or in another ViewDraw or other application window.

To copy a component and place it at a different location in the same window:

1. Click the component or object to select it.
2. Press the Ctrl key and click and drag the copied component to the new location in the active window. Release the mouse button to place it.

To copy a group of objects and place it at a different location in the same window:

1. Select the objects you want to copy.
2. Press and hold the Ctrl key while you click and drag the selected objects to the new location in the active window; release the mouse button to place them.

All selected objects are copied, maintaining their relative position.

To copy a component or object to another ViewDraw or another application window:

1. Click a component or click and drag over a group of components or objects to select them.
2. Choose Edit - Copy or click the button. Ctrl + C also copies the selected component or object.
3. Click on the window you want the component pasted in.
4. Choose Edit - Paste or click the button to paste the component or object. Ctrl+V also pastes the copied component or object.

To copy a bitmap picture of component other application window:

1. Click a component or click and drag over a group of components or objects to select them.
2. Choose Edit - Copy or click the button. Ctrl+C also copies the selected component or object.
3. Open and make active the application window that you want to paste the component in.
4. Choose Edit - Paste Special from the Edit menu or click the button to paste the component or object.
5. Click OK on the Paste Special dialog box.

Note: Bitmap images will show object as selected. If you want to copy a bitmap without the selection boxes, zoom in so that the section you want to copy fills the screen and perform the copy with nothing selected.

Adding Nets and Buses for Connectivity

What is a Net?

Use nets to specify an input or output lead from a component pin to any point on an existing net or bus. A net represents an electrical connection. You create nets between component pins, from a single component pin, or between nets.

You can construct a net with one or more segments. If a net has more than one segment, ViewDraw indicates the segment endpoints by joints at the net vertices. You can connect a maximum of four net segments at a net joint.

A net is not the same as a line. A line is only graphical; a net carries a signal.

What is a Bus?

Create buses between component bus pins, from a single component bus pin, or between nets. A bus is a collection of nets that can operate as a group or as individual nets within the bus.

A bus groups related signals. For example, the bus IN[0:7] represents the signals IN0, IN1, IN2, IN3, IN4, IN5, IN6, and IN7. You specify bus names and ranges (widths) using labels.

Note: Adding bus and net in the design requires labeling. When you attach bus or net to a component you have to label the bus and net, otherwise ViewDraw generates an error.

To add a bus to the active schematic:

1. Choose Add - Bus or click on the object toolbar. (You can also use the **b** hotkey or the **bus** command.)
2. Position the cursor at the originating point of the bus and click and drag to draw the bus. The current routing mode (straight, avoidance, or orthogonal) determines how the connection is formed.
3. Right-click to insert a vertex in the bus.

4. Release the mouse button to specify the ending point for the bus.

Once you enter Add Bus Mode, you can add multiple buses without selecting the Add Bus function again.

To add a net to the active schematic:

1. Choose Add - Net or click from the object toolbar.
2. Specify the originating point of the net. Click and drag to form the net, specifying the points along the net by right clicking. (By default, nets must begin at a component pin or at an existing net.) The current routing mode determines how the connection is formed.
3. Right-click or the spacebar with left mouse button depressed to insert a vertex in the net.
4. Release the left mouse button to specify an ending point for the net.

Once you enter Add Bus Mode, you can add multiple buses without selecting the Add Bus function again.

See Also

Schematic Guidelines

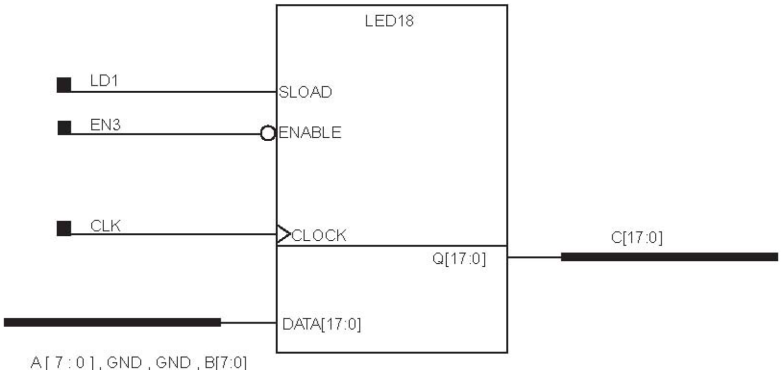
Bus Tips and Hints

Soft macros and ACTgen generated macros typically define data inputs as busses, for its symbols. There are several different ways to work with bus structures in ViewDraw for Actel.

Small to Larger

When busses (for instance combined with individual nets) are combined to generate a larger bus, the output and input busses cannot be physically connected on the schematic, as shown in Figure 5-15.

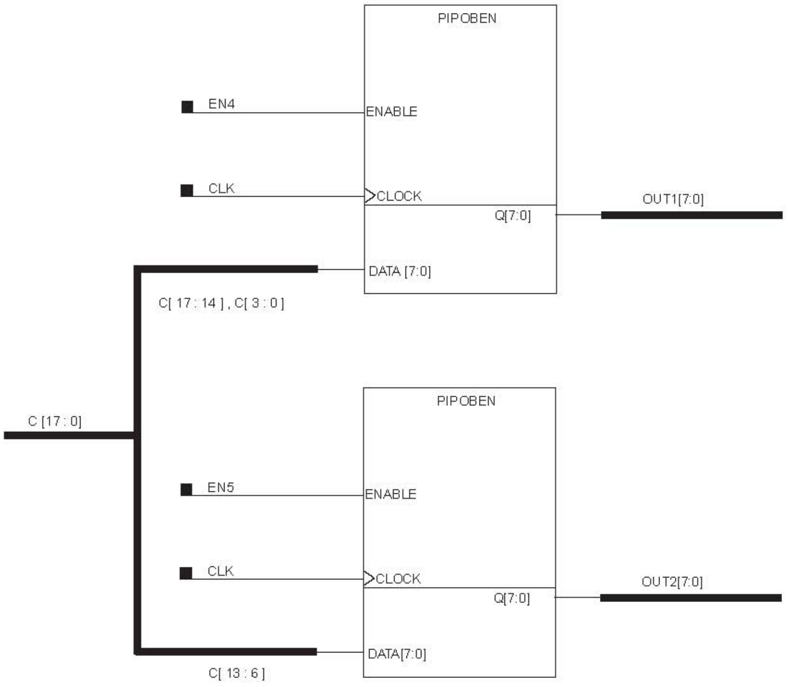
The net names map directly to the symbol names in the order defined (A7 = DATA17, first GND = DATA9, and B7=DATA7.) All inputs on a bus must be driven by some source even if the input is a "don’t care."



Small Bus Feeding Larger Bus

Large to Smaller

When a large bus feeds into smaller busses, the busses can be physically connected or separated depending on your preference. The inputs can be defined to have any bit sequence of the output and the label should be attached to the bus segment which directly attaches to the bus pin, as shown in Figure 5-16. Unused outputs will not create any errors (C4 and C5 are not used).



Large Bus Feeding Smaller Busses

Terminated to Ground or Terminated to VCC

A bus terminated all to ground or VCC can be defined by labeling the bus with the net names GND or VCC for each bit of the bus, or by using the "Constant Input Reduction" flag as follows.

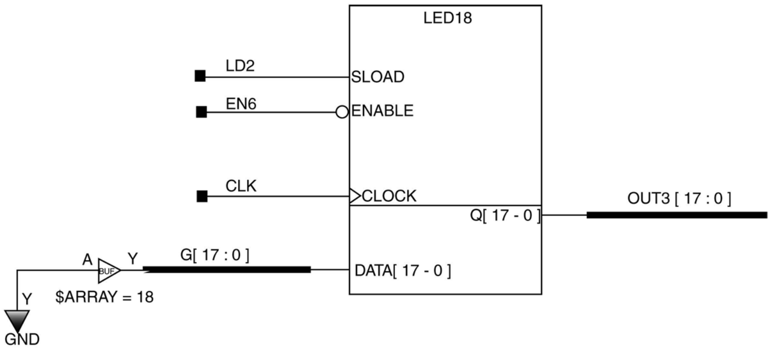
To use Constant Input Reduction:

1. Connect the GND or VCC symbol to the input of a buffer using a net.
2. Assign the buffer the following attribute:

$Array=bus_width (decimal value)

This attribute causes the buffer to be replicated in the netlist by the value of "bus_width."

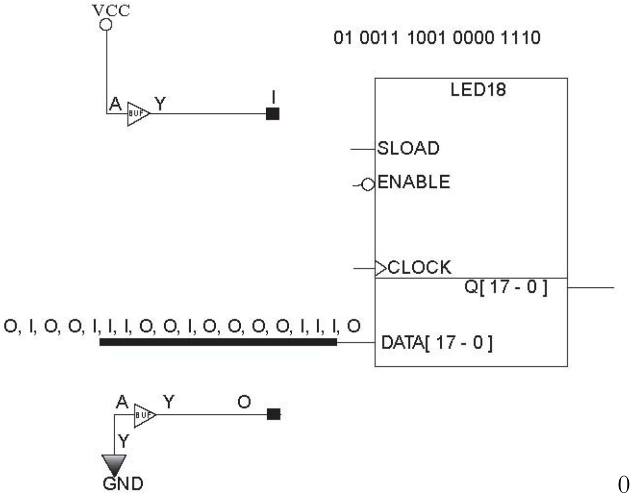
3. Label the bus with a name of appropriate width.



Constant Input Reduction

Terminated to Ground and VCC

A bus that has a constant value and is terminated to a combination of VCC and GND values can be defined in a number of ways. You can label the bus with the net names GND or VCC for each bit of the bus; however, for wide busses this can be very cumbersome. An alternative approach is to alias the VCC and GND names to single alphabetical character values, as shown below.



Alias Names

To use aliases:

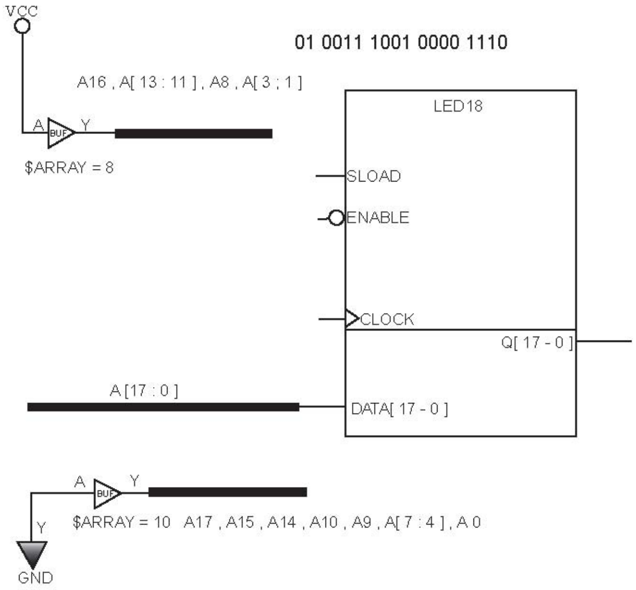
1. Connect GND and VCC symbols to the input of a buffer using a net.
2. Label the buffer output net with an alphabetic character. For example 1 (one) and O (zero).
3. Label the input bus using the same names for GND and VCC.

Another approach is to generate VCC and GND busses and then connect the appropriate value to each bus bit by giving the VCC and GND busses the same names as the input bus.

To use the input bus names:

1. Connect GND and VCC symbols to the input of a buffer using a net.
2. Assign the buffer the following attribute:

$ARRAY=bus_width(decimal_value)



Input Bus Names

This attribute causes the buffer to be replicated in the netlist by the value of "bus width."

3. Label the bus with the appropriate bits of the bus that feeds the input.

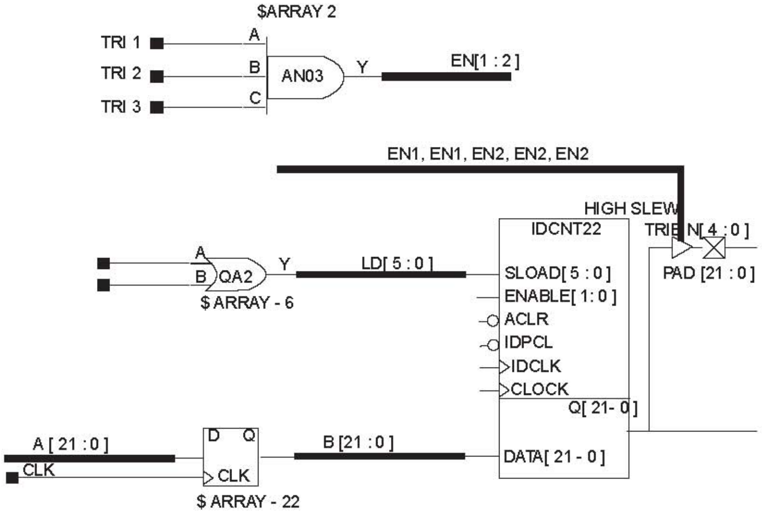
Logic Replication

There are some cases using the ACTgen Macro Builder where functional logic needs to be replicated to drive a bus.

The following example illustrates several options for replicating functional logic. In this example, the tristate enable pin "TRIEN[4:0]" is a 5-bit input bus. The TRIEN pin is controlled by a 3-input AND gate. Heavy loading for this case is not a problem, so the design uses two AND gates to drive the 5 TRIEN inputs. The synchronous load input pin "SLOAD[5:0]" is a 6-bit input bus which is controlled by a two-input pin OR gate. Heavy loading could be a problem for this case, so the design uses 6 OR gates.

The 22-bit register can be readily generated to drive the input data bus simply by applying the

$ARRAY=22 attribute.

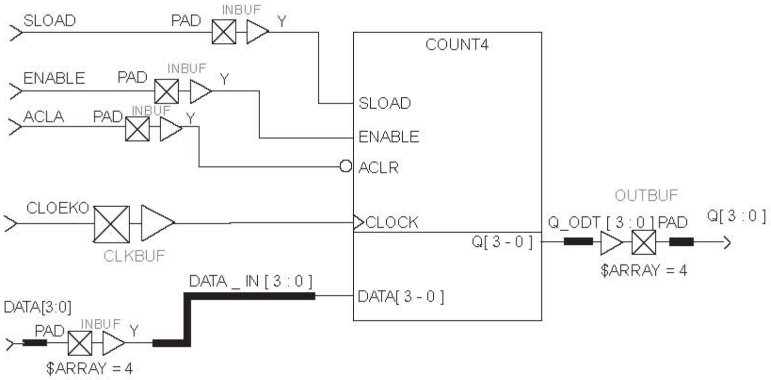


Logic Replication

Hierarchical Connectors

IN/OUT Hierarchical Connectors for Pure Schematic Designs

To preserve a bus port, both inbuf/outbuf from actelcells and in/out hierarchical connectors from builtin are required in top-level schematics, asshown in Figure 5-21 . Hierarchical connectors are always required for all lower level schematic blocks.



Hierarchical Connectors for Pure Schematic Designs

Hierarchy Connectors for Mixed Schematic-HDL Designs

For mix schematic-HDL designs, IN and OUT hierarchy connectors from the built-in library are required for ALL I/O ports in the top schematic. Actel I/O buffers are not needed at the top-level schematics, as I/O buffers will be automatically inserted during Synthesis. IN/OUT hierarchy connectors are always required in lower level schematics for symbol creation.

See Also

Bus Rippers

Re-Attaching Nets

A net may be removed from a component pin and dragged to a new component pin where it will re-attach.

To Re-attach a net:

1. Place the cursor over the pin end of the net segment you will be moving.
2. Click CTRL+r. This selects the net in move mode
3. Use the mouse to move the net to the new pin.
4. When the net is in place, click to release it.

Intersecting Connections

Two nets that cross make a connection only if a round solder dot appears at the crossing. Any incidental crossing of nets or buses from schematic edits does not imply a connection. Net solder dots appear at 3-way or 4-way connections. Bus solder dots appear only at 4-way connections.

To make an interconnection between two nets:

1. Choose Add - Net or click the ⊣ button.
2. Specify the originating point of the net and click and drag to form the net, specifying the points along the net by right clicking. The current routing mode determines how the connection is formed.
3. Right-click or the spacebar with left mouse button depressed to insert a vertex in the net.
4. Release the left mouse button to specify an ending point for the net.
5. Repeat steps 1 through 5 to create the intersecting net.
6. Choose Add - Net from the Add menu or click the ⊣ button again.
7. Click the mouse at the intersecting point of the nets to create the solder joint.

See Also

Adding Nets and Buses for Connectivity

Connecting Components

Connections by Abutment

Re-attaching Nets

Terminating a Connection

Dangling Connections

A net that does not connect to a pin or another net ends with a square box to indicate a dangling connection.

You can connect or continue a dangling net by clicking on the box while the Add Net command is activated.

Note: You can also select a segment and stretch it.

The Cleanup command from the Tools menu connects dangling nets to components and orthogonalizes nets as well.

Preserving Dangling Connections

To maintain dangling connectivity when you delete an object, you can hold down the **Ctrl** key when you select **Delete** from the Edit menu (or from the popup menu).

Adding Nets and Buses for Connectivity

What is a Net?

Use nets to specify an input or output lead from a component pin to any point on an existing net or bus. A net represents an electrical connection. You create nets between component pins, from a single component pin, or between nets.

You can construct a net with one or more segments. If a net has more than one segment, ViewDraw indicates the segment endpoints by joints at the net vertices. You can connect a maximum of four net segments at a net joint.

A net is not the same as a line. A line is only graphical; a net carries a signal.

What is a Bus?

Create buses between component bus pins, from a single component bus pin, or between nets. A bus is a collection of nets that can operate as a group or as individual nets within the bus.

A bus groups related signals. For example, the bus IN[0:7] represents the signals IN0, IN1, IN2, IN3, IN4, IN5, IN6, and IN7. You specify bus names and ranges (widths) using labels.

Note: Adding bus and net in the design requires labeling. When you attach bus or net to a component you have to label the bus and net, otherwise ViewDraw generates an error.

To add a bus to the active schematic:

1. Choose Add - Bus or click on the object toolbar. (You can also use the **b** hotkey or the **bus** command.)
2. Position the cursor at the originating point of the bus and click and drag to draw the bus. The current routing mode (straight, avoidance, or orthogonal) determines how the connection is formed.
3. Right-click to insert a vertex in the bus.
4. Release the mouse button to specify the ending point for the bus.

Once you enter Add Bus Mode, you can add multiple buses without selecting the Add Bus function again.

To add a net to the active schematic:

1. Choose Add - Net or click from the object toolbar.
2. Specify the originating point of the net. Click and drag to form the net, specifying the points along the net by right clicking. (By default, nets must begin at a component pin or at an existing net.) The current routing mode determines how the connection is formed.
3. Right-click on the spacebar with left mouse button depressed to insert a vertex in the net.
4. Release the left mouse button to specify an ending point for the net.

Once you enter Add Bus Mode, you can add multiple buses without selecting the Add Bus function again.

See Also

Schematic Guidelines

Connections by Abutment

ViewDraw automatically creates a net connection during component placement if the tip of an unconnected component pin contacts the tip of another unconnected component pin. The tip of the component pin is the point of pin contact with the symbol bounding box. You may connect pins that abut horizontally, vertically, or perpendicularly. Overlap of bounding boxes is allowed. Connection by abutment does not connect buses.

See Also

Adding Nets and Buses for Connectivity

Connecting Components By Abutment

Re-attaching Nets

Terminating a Connection

Connecting Components by Abutment

ViewDraw automatically creates a net connection during component placement if the tip of an unconnected component pin contacts the tip of another unconnected component pin. The tip of the component pin is the point of pin contact with the symbol bounding box. You may connect pins that abut horizontally, vertically, or perpendicularly. Overlap of bounding boxes is allowed. Connection by abutment does not connect buses.

ViewDraw attempts to connect abutting pins when these commands are used:

- Add Component
- Edit Copy
- Edit Paste
- Edit Array
- Edit Move

Connectivity

The net resulting from connection by abutment has all ordinary net characteristics, except its length is zero. The net exists in both the logical and graphical representations of the database. You can select the abutment net by clicking and dragging the left mouse button to define the selection area.

The Array command on the Edit menu makes connections between abutting pins during array creation (if a relative spacing of zero is used).

Restrictions

ViewDraw does not connect abutting pins on existing schematics when reading or writing. You must physically move the components to an abutting position on the existing schematic.

See Also

Adding Nets and Buses for Connectivity

Re-attaching Nets

Terminating a Connection

Adding Text

You can add text strings on symbols or schematics. Use these text strings to place notational text anywhere on the symbol or schematic. Text within a schematic or symbol sheet has no association with the graphical or connectivity data.

To add text to a symbol or schematic:

1. Choose Add - Text or click T on the object toolbar. (You can also use the **t** hotkey or the **text** command from the command line.)
2. Click and drag the mouse to move the text locator to the desired location.
3. Once you have the text locator where you want, release the mouse button.
4. Fill in the text Properties Dialog box. For more information about the fields in the Text Properties dialog box, click the Help button on the dialog box.

See Also

Creating a New Schematic

Creating a New Symbol

Creating a Pin Symbol

Creating a Sheet Border

Creating an Annotate Symbol

Creating Composite Symbols

Adding Graphics

You can add graphical objects to a symbol or schematic using the object toolbar buttons or the Add menu commands.

To add multiple graphical objects using Add Mode, select the Add command from the menu or toolbar while holding the **Ctrl** key.

To add an arc:

1. Choose Add - Arc or click ◜ on the Objects Toolbar.
2. Click to specify the first endpoint of the arc and drag the mouse to the location you select as the other endpoint for the arc.
3. Right-click or press the spacebar to specify the second endpoint.
4. Continue dragging to specify the arc midpoint.
5. Release left mouse button to finish arc.
6. To cancel arc placement, press Esc or release left mouse button before specifying the second endpoint for the arc.

To add a box:

1. Choose Add - Box or click □ on the object menu.
2. Click and drag to define the box.
3. Once you have the box you want, release the mouse button.

To add a circle:

1. Choose Add - Circle or click ○ on the object toolbar.
2. Click and drag the mouse to define the circle radius.
3. Once you have the circle you want, release the mouse button.

To add a line:

1. Choose Add - Line or click on the object toolbar.
2. Click and drag the mouse to define the line.
3. Right-click (or press spacebar) to create a polyline.
4. Once you have the line you want, release the mouse button.

See Also

Creating a New Schematic

Creating a New Symbol

Creating a Pin Symbol

Creating a Sheet Border

Creating an Annotate Symbol

Creating Composite Symbols

Creating Symbols

Creating a New Symbol

When creating any type of symbol, you graphically construct the symbol body using the Arc, Box, Circle, and Line commands.

Note: Do not use lines to create pins. Use the Pin command or P button. (You can also use the p hotkey or the pin command from the command line.)

You can create module, composite, pin, or annotate symbols.

- A module symbol does not have an underlying schematic. Module symbols represent a base function in the design.

Click here for instruction on creating a module symbol.

- A composite symbol has an underlying schematic. The symbol implements the underlying schematic function at a higher level of the design.

Click here for instruction on creating a composite symbol.

- A pin symbol represents a port or interface on the schematic (for example, IN, OUT, or BI built-in symbols). You also use pin symbols to tie a net to a global symbol.

Click here for instructions on creating a pin symbol.

- An Annotate symbol represents graphics or annotation that can be put on a schematic, but has no electrical or connectivity information.

Click here for instructions on creating an annotate symbol.

See Also

Creating a Pin Symbol

Creating a Sheet Border

Creating an Annotate Symbol

Creating Analog Symbols

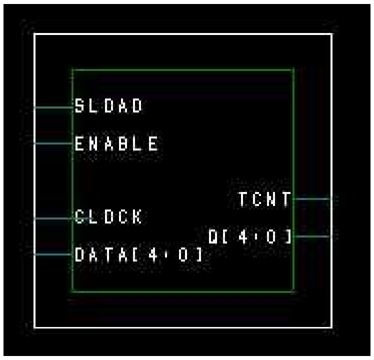
Creating Composite Symbols

Creating the Underlying Schematic for an Analog Symbol

Creating VHDL and Verilog Symbols

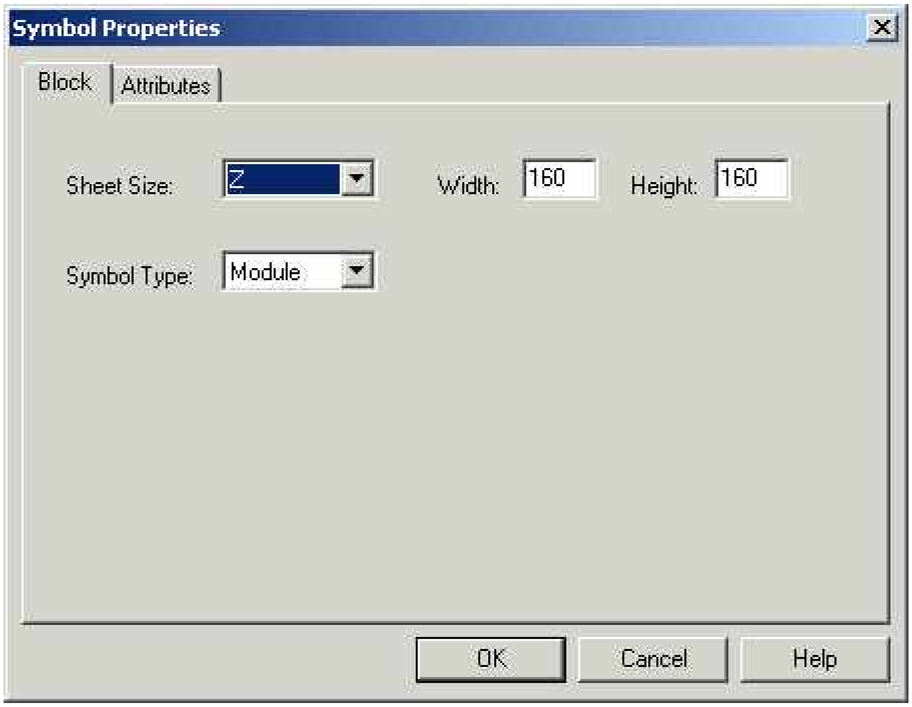
Resizing Existing Symbols

From the top level, right-click the symbol you want to resize and select **Symbol**. This zooms into a symbol icon. You should see two boxes, an inner box and an outer box.



To resize the outer box:

1. Right-click the outer box and select **Properties**. The **Symbol Properties** dialog box appears.
2. Enter the desired width and height.



Sheet Size: Z

Width: 160

Height: 160

Symbol Type: Module

To resize the inner box:

1. Left-click the inner box to select it (it should change color).
2. From the menu, select **Edit - Stretch**.
3. Drag and resize the inner box.

Note: Drag the pin wires to adjust them to the desired length.

Adding Graphics

You can add graphical objects to a symbol or schematic using the object toolbar buttons or the Add menu commands.

To add multiple graphical objects using Add Mode, select the Add command from the menu or toolbar while holding the **Ctrl** key.

To add an arc:

1. Choose Add - Arc or click on the Objects Toolbar.
2. Click to specify the first endpoint of the arc and drag the mouse to the location you select as the other endpoint for the arc.
3. Right-click or press the spacebar to specify the second endpoint.
4. Continue dragging to specify the arc midpoint.
5. Release left mouse button to finish arc.
6. To cancel arc placement, press Esc or release left mouse button before specifying the second endpoint for the arc.

To add a box:

1. Choose Add - Box or click □ on the object menu.
2. Click and drag to define the box.
3. Once you have the box you want, release the mouse button.

To add a circle:

1. Choose Add - Circle or click ○ on the object toolbar.
2. Click and drag the mouse to define the circle radius.
3. Once you have the circle you want, release the mouse button.

To add a line:

1. Choose Add - Line or click ∿ on the object toolbar.
2. Click and drag the mouse to define the line.
3. Right-click (or press spacebar) to create a polyline.
4. Once you have the line you want, release the mouse button.

See Also

Creating a New Schematic

Creating a New Symbol

Creating a Pin Symbol

Creating a Sheet Border

Creating an Annotate Symbol

Creating Composite Symbols

Adding Text

You can add text strings on symbols or schematics. Use these text strings to place notational text anywhere on the symbol or schematic. Text within a schematic or symbol sheet has no association with the graphical or connectivity data.

To add text to a symbol or schematic:

1. Choose Add - Text or click T on the object toolbar. (You can also use the **t** hotkey or the **text** command from the command line.)
2. Click and drag the mouse to move the text locator to the desired location.
3. Once you have the text locator where you want, release the mouse button.
4. Fill in the text Properties Dialog box. For more information about the fields in the Text Properties dialog box, click the Help button on the dialog box.

See Also

Creating a New Schematic

Creating a New Symbol

Creating a Pin Symbol

Creating a Sheet Border

Creating an Annotate Symbol

Creating Composite Symbols

Adding Pins to a Symbol

A pin provides a connection point for nets that carry signals into or out of a component.

To add a pin to a symbol:

1. Choose the Add -- Pin command or click the P button. (You can also use the **p** hotkey or the **pin** command from the command line.)
2. Click and drag the cursor to form the pin and release the left mouse button.

The pin snaps to the nearest edge of the border.

3. Label the Pin.
4. Add pin numbers and Attributes to the Pin.

Note: Do not use a line to create a pin! A pin establishes connectivity between components; a line represents only a graphical object.

Things to consider during pin placement:

- Be consistent in pin spacing.

Actel recommends a 0.2 inch spacing to match the pin spacing used on symbols in the standard libraries.

- Place pins every other grid space on a grid with spacing of 10 (the default).
- This allows straight, on-grid connections, regardless of the type of routing mode you use.

See Also

Adding Attributes

Adding Labels

Adding Pin Numbers

Attached vs. Unattached Attributes

Attribute Format

Attribute Ranges

AutoIncrementing Labels

Changing the Pin Order

Compound Labels

Creating a Pin Order

Distinguishing Labels from Attributes

Expanding Pin Attributes

Inverting Pins

Label Ranges

Label Scope

Marking Unused Pins

Multiple and Duplicate Attributes

Multiple Attribute Values

Tools -- Save Pins to file command

User-Defined Attributes

Visibility Options

Adding Pin Numbers

You specify pin numbers using the # attribute. The format for setting the # attribute for a pin is as follows:

- Attribute Name #
- Attribute Value *n,n,n*

Once you place a symbol on a schematic, you cannot change the pin numbers by modifying the # attributes on the component level.

To change pin numbers after they are placed on the schematic

1. Select the component.
2. Execute the **Slot** command from the **Edit** menu. This reflects the changes to the # and REFDES attributes made to the symbol.

See Also

Adding Attributes

Attached vs. Unattached Attributes

Attribute Format

Attribute Ranges

Distinguishing Labels from Attributes

Expanding Pin Attributes

Multiple and Duplicate Attributes

Multiple Attribute Values

User-Defined Attributes

Visibility Options

Expanding Pin Attributes

ViewDraw attributes are normally expanded when added via the Properties box. Here is a more automated method that can be used to when many pins must have an expanded attribute.

If Pin Numbers are sequential:

1. Choose Add - Pin or click to add a pin in the symbol window. (You can also use the **p** hotkey or the **pin** command from the command line.)
2. Double-click the pin and choose the Attribute Tab.
3. Add attribute #=1 and select **Set** and **OK**. (Arrange placement as required.)
4. Choose Add - Array. Enter these settings in the Add Array dialog box: 4 Rows, 1 Column, 10 Row Spacing. (You will note that each of the four pins has the attribute #=1.)
5. Choose Edit - Replace and select Attribute as the object type.
6. Enter #=1 in the Expression box and #=[1:4] in the Replace box. 7.
7. Click **Select** and then **Replace**.

If Pin Numbers are not sequential:

1. Choose Add - Pin or click to add a pin in the symbol window. (You can also use the **p** hotkey or the **pin** command from the command line.)
2. Double-click the pin and choose the Attribute Tab.
3. Add attribute #=1 and select **Set** and **OK**. (Arrange placement as required.)
4. Choose Add - Array. Enter these settings in the Add Array dialog box: 4 Rows, 1 Column, 10 Row Spacing. (You will note that each of the four pins has the attribute #=1.)
5. Select pin one with a left click, then pins 2,3,4... with a CTRL + Click.
6. Choose Edit - Replace and select Attribute as the object type.

<Selected attributes> appears in the **Expression** box. Enter #=[4:1] in the **Replace** box. Note the Selected Attributes are replaced as a LIFO (Last In First Out) sequence.

7. Click **Replace**.

See Also

Adding Attributes

Adding Pin Numbers

Adding Pins to a Symbol

Attached vs. Unattached Attributes

Attribute Format

Attribute Ranges

Changing the Pin Order

Distinguishing Labels from Attributes

Inverting Pins

Marking Unused Pins

Multiple and Duplicate Attributes

Multiple Attribute Values

Tools -- Save Pins to file command

User-Defined Attributes

Visibility Options

Marking Unused Pins

If you want to mark unused pins to prevent unwanted error messages from the PCB netlist report, follow these steps:

1. Add the following in the general section of the attribute pass list of the PCB configuration file:

NET NC

2. Add a net stub to any unconnected pin and place the NC attribute with no value on it.

This removes the Warning 6094 Pin not connected, for all pins with this attributed net stub attached. Once you add the stub to one pin, it can be easily copied to any others using Control Drag method for on sheet copies or Cut and Paste for off sheet copies.

See Also

Adding Pin Numbers

Adding Pins to a Symbol

Changing the Pin Order

Creating a Pin Order

Inverting Pins

Tools -- Save Pins to file command

Creating a Pin Order

A pin order is the list of pin connections in the wirelist file. The system determines the pin order in these three phases (in order or priority):

A PINORDER symbol attribute

PINORDER is a symbol attribute. For more information about this attribute, refer to the attribute glossary.

A pin order file

A pin order file is a text file, with a .pin file extension, located in the sym subdirectory. The .pin file contains the pin labels or all symbol pins -- not just bus pins. There is no limit to the number of pins you can specify in this file.

An alphabetical ordering of the symbol’s pin labels

If no PINORDER attribute or pin order file exists, the wirelist makes a list of the symbol’s pin labels. The wirelist sorts the list alphabetically and uses it as the pin order.

When replacing or adding a Built-in module, the pin order must be identical to the pin order as defined for ViewSim.

To create a PINORDER attribute:

1. With nothing selected, right-click and choose **Properties** from the shortcut menu.
2. Select the Attributes tab from the Properties dialog box.
3. Enter PINORDER in the Name field and enter the pin labels in the Value field.
4. Click the **Set** button to set the attribute.
5. Click **OK** to accept the change and dismiss the dialog box.

To create a pin order file:

1. Using any text editor, create a file using the .pin file extension in the sym subdirectory.
2. List the pin labels in the appropriate pin order, separating the pin labels with a space (or place them on separate lines).
3. Save and exit the file.

Refer to the Attribute Glossary for more information on the PINORDER Attribute.

Tip: If a pin already has a PINORDER attribute, you can use the Tools -- Save Pins to file command to make the pin order file.

See Also

Adding Pin Numbers

Adding Pins to a Symbol

Changing the Pin Order

Inverting Pins

Marking Unused Pins

Tools -- Save Pins to file command

Changing the Pin Order

The method you use to change the pin order depends on the present form of the pin order specification.

To change pin orders set by attributes:

1. Double-click the attribute to select it.
2. Change the pin labels in the Value field.
3. Click the **Set** button to set the attribute.
4. Click **OK** to accept the change and dismiss the dialog box.

To change pin orders set by a .pin file:

1. Using any text editor, open the .pin file you want to update.
2. Change or add pin labels in the appropriate pin order, separating the pin labels with a space (or place them on separate lines).
3. Save and exit the file.

Pins on the symbol and in the schematic must match the pin labels listed in the pin order. Otherwise, errors result while checking and wirelisting.

See Also

Adding Pin Numbers

Adding Pins to a Symbol

Creating a Pin Order

Inverting Pins

Marking Unused Pins

Tools -- Save Pins to file command

Adding Attributes

You use attributes to compose symbol definitions for interpretation when wirelisting. The wirelist uses the first symbol definition it encounters and interprets the attributes associated with this symbol. Therefore, you should add attributes at hierarchical levels where the wirelist will use the definition that you intend.

You can add attributes to:

- Symbols (unattached)
- Symbol pins
- Schematics (unattached)
- Components
- Component pins
- Net Segments
- Bus segments

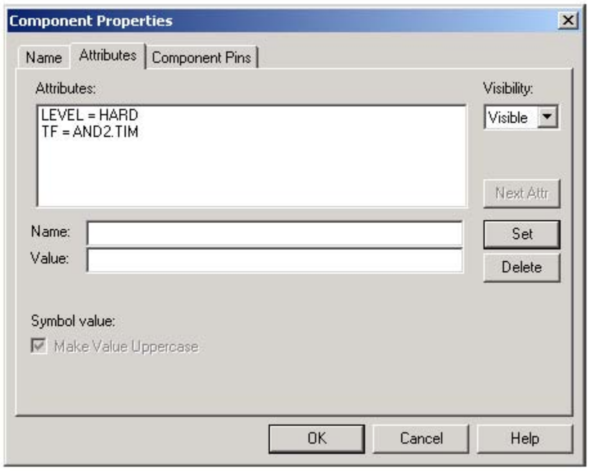
You cannot place attributes on boxes, lines, arcs or circles.

To add an attribute:

1. Double-click the object you want to add the attribute to.

The Properties dialog box appears.

2. Click the **Attributes** tab of the dialog box.



LEVEL = HARD
TF = AND2.TIM

☑ Make Value Uppercase

Component Properties Dialog Box

3. Enter the attribute name in the **Name** field.
4. Enter the attribute value in the **Value** field.
5. Select the attribute visibility selection from the **Visibility** pull-down list.

Note: Set the REFDES attribute on symbols and the # attribute on symbol pins to invisible because their values are automatically promoted to the component level on the schematic.

6. Select the **Set** button to add the attribute.
7. Click **OK** to close the **Properties** dialog box.

See Also

Adding Pin Numbers

Adding Pins to a Symbol

Attached vs. Unattached Attributes

Attribute Format

Attribute Ranges

Changing the Pin Order

Creating a Pin Order

Distinguishing Labels from Attributes

Expanding Pin Attributes

Inverting Pins

Marking Unused Pins

Multiple and Duplicate Attributes

Multiple Attribute Values

Tools -- Save Pins to file command

User-Defined Attributes

Visibility Options

Changing the Block Type of New Symbols

You can change the block type for new symbols by using the Project Settings dialog box.

To change the block type for new symbols:

1. Choose Project - Settings.
2. Select the Block tab on the Project Settings dialog box.
3. Change the symbol type option to the correct block type.
4. Click **OK** to save the settings and dismiss the Project Settings dialog box.

To change the block type for existing symbols:

1. Open the symbol.
2. Double-click a blank area of the symbol to get the Symbols Properties dialog box.
3. Select the desired block type.
4. Click **OK** to accept the change and dismiss the dialog box.

Creating Composite Symbols

Creating Composite Symbols

To open the symbol window:

1. Open a symbol window using the File - Open. The **Open** dialog box appears.
2. Select **Symbol** from the **Type** field of the **Open** dialog box.
3. Enter a new name for the symbol in the **Symbol** field of the **Open** dialog box.
4. Click **OK** to open the new symbol window.

To set up the symbol settings:

The default symbol type is composite so you do not need to change the symbol settings unless you want to change the sheet size of the symbol.

To change the sheet size:

1. Right-click and select **Properties** from the shortcut menu or double-click an empty space in the symbol. The **Properties** dialog box appears.
2. Select the **Block** tab on the **Properties** dialog box.
3. Select a new sheet size from the **Sheet Size** option.
4. Click **OK** to save the settings.

To complete the symbol:

1. Create the symbol body using the Arc, Box, Circle, or Line, commands from the **Add** menu or click the appropriate toolbar button.
2. Add the Pins to the symbol using the Add - Pin command or click the button. (You can also use the **p** hotkey or the **pin** command from the command line.)

3. Add labels to the pins.
4. Add attributes to the symbol.
5. Select the File - Save command to save the symbol.

See Also

Adding Attributes

Adding Graphics

Adding Labels

Adding Pin Numbers

Adding Pins to a Symbol

Attached vs. Unattached Attributes

Attribute Format

Attribute Ranges

AutoIncrementing Labels

Changing the Pin Order

Compound Labels

Creating a Pin Order

Creating a Sheet Border

Distinguishing Labels from Attributes

Expanding Pin Attributes

Inverting Pins

Label Ranges

Label Scope

Marking Unused Pins

Multiple and Duplicate Attributes

Multiple Attribute Values

Tools -- Save Pins to file command

User-Defined Attributes

Visibility Options

Creating Module Symbols

Creating a module symbol requires several steps.

To open the symbol window:

1. Open a symbol window using the File - Open command. The **Open** dialog box appears.
2. Select **Symbol** from the Type field of the Open dialog box.
3. Enter a new name for the symbol in the **Symbol** field of the **Open** dialog box.

4. Click **OK** to open the new symbol window.

To set up the symbol settings:

1. Right-click and select **Properties** from the shortcut menu or double-click an empty space in the symbol. The **Properties** dialog box appears.
2. Select the **Block** tab on the **Properties** dialog box.
3. Change the **Symbol Type** to **Module**.
4. Change the sheet size if necessary.
5. Click **OK** to save the settings.

Completing the symbol:

1. Create the symbol body using the Arc, Box, Circle, or Line, commands from the **Add** menu or click the appropriate toolbar button.
2. Add the Pins to the symbol using the Add - Pin command or click the button. (You can also use the **p** hotkey or the **pin** command from the command line.)
3. Add labels and attributes to the pins.
4. Add attributes to the symbol.
5. Select the File - Save command to save the symbol.

See Also

Adding Attributes

Adding Graphics

Adding Labels

Adding Pin Numbers

Adding Pins to a Symbol

Attached vs. Unattached Attributes

Attribute Format

Attribute Ranges

AutoIncrementing Labels

Changing the Pin Order

Compound Labels

Creating a Pin Order

Creating a Sheet Border

Distinguishing Labels from Attributes

Expanding Pin Attributes

Inverting Pins

Label Ranges

Label Scope

Marking Unused Pins

Multiple and Duplicate Attributes

Multiple Attribute Values

Tools -- Save Pins to file command

User-Defined Attributes

Visibility Options

Creating an Annotate Symbol

Creating an annotate symbol requires several steps.

To open the symbol window:

1. Open a symbol window using the File - Open. The **Open** dialog box appears.
2. Choose **Symbol** from the **Type** field of the **Open** dialog box.
3. Enter a new name for the symbol in the **Symbol** field of the **Open** dialog box.
4. Click **OK** to open the new symbol window.

To set up the symbol settings:

1. Right-click and select **Properties** from the shortcut menu or double-click an empty space in the symbol. The **Properties** dialog box appears.
2. Select the **Block** tab on the **Properties** dialog box.
3. Change the **Symbol Type** to **Annotate** in the **Project Settings** dialog box.
4. Change the sheet size if necessary.
5. Click **OK** to save the settings.

Completing the symbol:

1. Create the symbol body using the Arc, Box, Circle, or Line, commands from the **Add** menu or click the appropriate toolbar button.
2. Add labels to the objects.
3. Select the File - Save command to save the symbol.

See Also

Adding Attributes

Adding Graphics

Adding Labels

Adding Pin Numbers

Adding Pins to a Symbol

Attached vs. Unattached Attributes

Attribute Format

Attribute Ranges

AutoIncrementing Labels

Changing the Pin Order

Compound Labels

Creating a Pin Order

Creating a Sheet Border

Distinguishing Labels from Attributes

Expanding Pin Attributes

Inverting Pins

Label Ranges

Label Scope

Marking Unused Pins

Multiple and Duplicate Attributes

Multiple Attribute Values

Tools -- Save Pins to file command

User-Defined Attributes

Visibility Options

Creating a Pin Symbol

Creating a pin symbol requires several steps.

To open the symbol window:

1. Open a symbol window using the File - Open. The **Open** dialog box appears.
2. Choose **Symbol** from the **Type** field of the **Open** dialog box.
3. Enter a new name for the symbol in the **Symbol** field of the **Open** dialog box.
4. Click **OK** to open the new symbol window.

To set up the symbol settings:

1. Right-click and select **Properties** from the shortcut menu or double-click an empty space in the symbol. The **Properties** dialog box appears.
2. Select the **Block** tab on the **Properties** dialog box.
3. Change the **Symbol Type** to **Pin** in the **Project Settings** dialog box.
4. Change the sheet size if necessary.
5. Click **OK** to save the settings.

Completing the symbol:

1. Create a sheet border for the symbol.
2. Create the symbol body using the Arc, Box, Circle, or Line, commands from the Add menu or click the appropriate toolbar button.
3. Add the Pins to the symbol using the Add - Pin or click the P button. (You can also use the **p** hotkey or the **pin** command from the command line.)

4. Add labels to the pins.
5. Add attributes to the symbol.
6. Select the File - Save command to save the symbol.

Note: If you recreate the symbol, you must close the Viewdraw session from File – Exit. Right-click the module in Design Hierarchy and click Create Symbol.

See Also

Adding Attributes

Adding Graphics

Adding Labels

Adding Pin Numbers

Adding Pins to a Symbol

Attached vs. Unattached Attributes

Attribute Format

Attribute Ranges

AutoIncrementing Labels

Changing the Pin Order

Compound Labels

Creating a Pin Order

Creating a Sheet Border

Distinguishing Labels from Attributes

Expanding Pin Attributes

Inverting Pins

Label Ranges

Label Scope

Marking Unused Pins

Multiple and Duplicate Attributes

Multiple Attribute Values

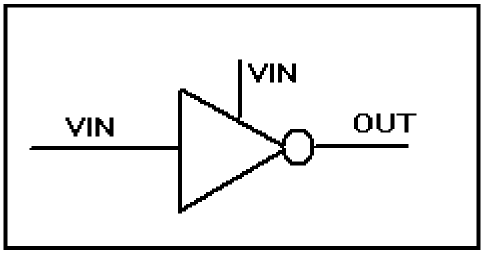
Tools -- Save Pins to file command

User-Defined Attributes

Visibility Options

Creating Analog Symbols

Creating analog symbols in ViewDraw can be very simple or very complex. This procedure that follows will help you create a fully operational and simulatable analog component. The following steps will help you create the following CMOS inverter.



To create an analog symbol:

1. Choose File > Open.
2. Select **Symbol** as the **Type** and enter a new name in the **Symbol** text box.
3. Click **OK** to create the Symbol window.
4. Draw the symbol's triangular shape using the Line command from the **Add** menu (or the button).
5. Add the pins using the Pin command from the **Add** menu (or the button).

Pins establish points of connection between your symbol shape and outside components.

6. Select the pin and double-click to choose the Properties option.
7. Select the **Name** tab of the **Properties** dialog box and label the pins.

Remember that labeling the pins establishes unique connectivity across hierarchical levels. For more information on labeling pins, refer to Adding Labels.

8. Change the pin sense by checking the **Inverted Pin** check box.
9. Choose **File - Save** to save the symbol.

See Also

Creating Analog Symbols

Changing Properties

Properties and Settings

You can change the properties associated with an entire project or you can change just the properties associated with an individual component, object, or symbol and schematic sheet.

Individual properties differ for each object type. Project properties effect the entire project.

Object Properties

Object	Properties	Object	Properties
Component	Label	Pin	Label
	Attributes		Attributes
	Component Pin Attributes		Inversion
Graphical Object	Color	Net	Label
	Line Style		Attributes
	Fill Style	Text (Annotation)	Text String
Symbol Sheet	Block Properties		Size
	Sheet Size		Origin
	Attribute Names		Color, Line, Font
	Attribute Values	Attribute	Text
	Attribute Visibility		Visibility
Schematic Sheet	Block Properties	Label	Text
	Sheet Size		
	Attribute Names		Scope
	Attribute Values		Visibility
	Attribute Visibility		Invert

Project Properties and Settings

Category	Settings			
Project	Selection Distance	Grid	Detail	ExtraErrs
	Grid Spacing	Border	Values	Course Grid
	Autolog	Header	Names	OATs
	Undo	Rippers	Sort	Verbose Err
Block	Sheet Size	Symbol Type	Default Sheet	
Components	Display Component Text	Display Pin Numbers	Display REFDES	Symbol Qualifiers
Nets	Route Mode	Avoidance Distance	Snap to Pin	Global Net Names
	Thick Nets	Dot Size	Box Size	Bus Dot Size
	Bus Width			
Attributes	Display Attributes	New Attribute Visibility	Attribute List	Name
	Value			
Labels	Display Labels	Unique Label on Copy	Scope -- Global or Local	
Text	Display Text	Display Threshold	Size	Origin
Colors	Object	Color	Line Style	Fill Style

See Also

Changing Individual Component or Object Properties

Changing Project Properties and Settings

Changing Project Properties and Settings

To change properties for the overall project:

1. Choose the **Project - Settings** command.
2. Change project setting properties by selecting the appropriate tab and making your selections.
3. Click **OK** to accept the changes.

See Also

Changing Individual Component or Object Properties

Properties and Settings

Changing Individual Component or Object Properties

To change properties for an individual component, object, symbol sheet, or schematic sheet:

1. Select the object, component, or sheet.
2. Right-click and select **Properties** from the shortcut menu.
3. Change properties by selecting the appropriate tab and making you selections.
4. Click **OK** to accept the changes.

See Also

Changing Project Properties and Settings

Properties and Settings

Adding Labels and Attributes

Adding Labels

Labels are strings used to uniquely identify nets, buses, and components. Components and Pins have only one label. Nets and buses can have multiple labels of the same name -- one label per net or bus segment.

If you do not label components, pins, nets, or buses, ViewDraw automatically assigns an internal name to these objects. All internal names begin with $ (for example, $1P12). Once you label an object, ViewDraw displays the label you specify in addition to the internal name.

All labels have:

- -- not inverted or inverted
- visibility -- visible or invisible
- sense scope -- local or global

ViewDraw maintains connectivity between a symbol and the underlying schematic through pin and net labels. You must label all pins on a symbol and you must label the nets of the underlying schematic to correspond to the pins on its symbol.

Pin Labels

You specify pin names and ranges (width) with labels. Pin labels establish and maintain connectivity between components. The pin range of a multi-bit pin must be identical to the range of a connecting bus.

Note: Except when the component has the $ARRAY attribute.

All pins on a symbol must have a label for identification during wirelisting. For symbols that do not normally have labels, such as resistors and capacitors, make the labels invisible.

For more information on Label Ranges, go to the Label Ranges help section.

Bus Labels

Bus labels specify the nets contained in the bus. The labels of the nets connected to the bus must correspond to the bus label. For example, a bus labeled XBUX[1:3] can have only nets labeled XBUS1, XBUS2, and XBUS3.

To add a label to an object:

1. Select the object you want to label.
2. Double-click the object (or right-click and select **Properties** from the shortcut menu).
3. Enter a label name and range in the Label field of the Properties dialog box. ViewDraw translates all labels to uppercase if you do not enter them that way.
4. Make label and visibility selections.
5. Make label scope selections.
6. Click **OK** to accept the label settings.

Easy labeling of busses or nets:

After you initially label a bus or net, you can easily label subsequent instances, including nets or busses that you have extracted from the original instances.

ViewDraw gives you lists of labels for different categories of objects. You select one of these lists from the **Label From** popup. You can also control the format of the lists using the Display Threshold.

- All Bus Elements - Every label associated with the selected bus.
- Unused Bus elements - Unassigned labels associated with the selected bus.
- In Memory Labels - All labels associated with the currently open schematic or symbol.
- MRU Labels - The last 20 labels you have used on any object.
- Connected Symbol Pins - Labels for symbol pins that are connected to the selected net on a schematic.
- All Symbol I/O - Labels for all symbol pins on the symbol associated with the currently open schematic.
- Unused Symbol I/O - Unassigned labels for all symbol pins on the symbol associated with the currently open schematic.

To use previously assigned labels:

1. Right-click the net or bus you want to label.
2. From the shortcut menu, select **Label From**
3. From the shortcut menu, select which label list you want to use. ViewDraw opens a dialog box containing the label list you have selected.
4. Select the label you want to use. If you are labeling a bus, you can select multiple labels by using the **SHIFT** and **CTRL** keys.

To set the list display threshold:

By default, ViewDraw displays label lists in a dialog box. If you prefer, you can display label lists in a popup menu. You specify a threshold value below which a popup displays and above which a dialog box displays.

1. From the **Project** menu, click **Settings**, and then click the **Labels** tab.
2. In the **Mystery Number** box, enter the threshold you want to set.

If the number of labels available is less than the threshold, they will be displayed as a popup. If the number is greater than or equal to the threshold, they will be displayed in a dialog box.

See Also

AutoIncrementing Labels

Compound Labels

Distinguishing Labels from Attributes

Label Ranges

Label Scope

Label Ranges

Labels can express a numerical range for a bus or a pin by specifying beginning, ending, and incremental integer values. ViewDraw expands labels with a range to create a series of labels.

Label Range Format:

The format for label ranges is: NAME[F:L:I] where F is the first number, L is the last number, and I is the interval between the numbers in the range. If no interval is specified, the interval is 1.

Example:

out.[1:5] creates out.1, out.2, out.3, out.4, and out.5

a[10:3:2] creates a10, a8, a6, a4

a[1:2]b[2:1] creates a1b2, a1b1, a2b2, a2b1

A radix qualifier may follow the range specification to indicate a binary, hexadecimal, or octal specification. Separate the qualifier from the range specification with a backslash.

Example:

The label A[8:C\H] expands to create A8, A9, AA, AB, and AC.

Restrictions:

You can specify a bus range only with numeric characters. For example, data[b7:b0] is not a valid format. If the bus range you want to label is decimal (for example, 7:0), use the format data[7:0].

See Also

Adding Labels

AutoIncrementing Labels

Compound Labels

Distinguishing Labels from Attributes

Label Scope

AutoIncrementing Labels

You can automatically increment, or deincrement, component, net, or pin labels. You can autoincrement labels on individually selected objects, or you can select an object and autoincrement labels as you copy or array the object.

To autoincrement labels:

1. Select the component, pin, or net segment.
2. Double-click the selected object to display the **Properties** dialog box.
3. Enter the label in the **Label** field of the **Name** Tab. Specify the label as a range.
4. Click **OK** to accept the label for selected component, pin, or net.
5. Select the next object to label.
6. Expand the label using Copy or Array commands, or the **Next** Label button on the **Properties** dialog box.

Example of Autoincrement Without Expand:

This example places and labels 4 NAND gate with four quick steps:

1. Add a NAND component using the Add - Component command.
2. Double-click the component to display the Name Tab on the Properties dialog box.
3. Specify the label increment N[1:4] in the Label field.
4. Execute the Copy command from the Edit menu three times. The labels N2, N3, and N4 appear in successions as you place the components.

See Also

Adding Labels

Compound Labels

Distinguishing Labels from Attributes

Label Ranges

Label Scope

Label Scope

Label scopes can be global or local.

Global labels are known throughout the hierarchy. Flattened wirelists interpret global labels. However, hierarchical wirelist interpretation depends on the capability of the target system (for example, HSPICE supports global labels).

Local labels are only known within the schematic at one level in the hierarchy.

You can selectively override the local default for a specified label using the NETNAME keyword in the viewdraw.ini file. This is added by specifying the global Net Name on the Nets Tab of the Project Settings dialog box.

See Also

Adding Labels

AutoIncrementing Labels

Compound Labels

Distinguishing Labels from Attributes

Label Ranges

Compound Labels

Use compound labels to label buses or pins that are not expressible as a single range. You form a compound label with several label names (with or without ranges). You separate the labels using commas. The system expands compound labels separately from left to right.

To enter a compound label, use the following format:

NAME[F:L:I], NAME,NAME[F:L],... where F is the first number, L is the last number, and I is the interval between the numbers in the range. If no interval is specified, the interval is 1.

Examples:

A[0:3],B,D[1:4] = A0,A1,A2,A3,B,D1,D2,D3,D4

A[2:1],B[3:0],C = A2,A1,B3,B2,B1,B0,C

See Also

Adding Labels

AutoIncrementing Labels

Distinguishing Labels from Attributes

Label Ranges

Label Scope

Adding Attributes

You use attributes to compose symbol definitions for interpretation when wirelisting. The wirelist uses the first symbol definition it encounters and interprets the attributes associated with this symbol. Therefore, you should add attributes at hierarchical levels where the wirelist will use the definition that you intend.

You can add attributes to:

- Symbols (unattached)
- Symbol pins
- Schematics (unattached)
- Components
- Component pins
- Net Segments
- Bus segments

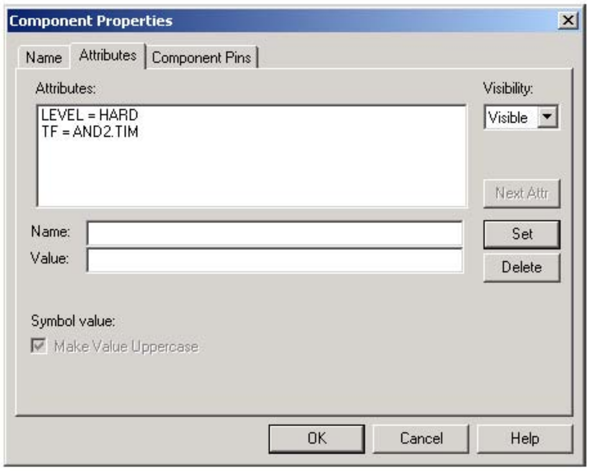
You cannot place attributes on boxes, lines, arcs, or circles.

To add an attribute:

1. Double-click the object you want to add the attribute to.

The Properties dialog box appears.

2. Click the **Attributes** tab of the dialog box.



Attributes:
LEVEL = HARD
TF = AND2.TIM

Visibility: Visible

Name:

Value:

Symbol value:

☑ Make Value Uppercase

Component Properties Dialog Box

3. Enter the attribute name in the **Name** field.

4. Enter the attribute value in the **Value** field.

5. Select the attribute visibility selection from the **Visibility** pull-down list.

Note: Set the REFDES attribute on symbols and the # attribute on symbol pins to invisible because their values are automatically promoted to the component level on the schematic.

6. Select the **Set** button to add the attribute.

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

See Also

Adding Pin Numbers

Adding Pins to a Symbol

Attached vs. Unattached Attributes

Attribute Format

Attribute Ranges

Changing the Pin Order

Creating a Pin Order

Distinguishing Labels from Attributes

Expanding Pin Attributes

Inverting Pins

Marking Unused Pins

Multiple and Duplicate Attributes

Multiple Attribute Values

Tools -- Save Pins to file command

User-Defined Attributes

Visibility Options

Attached vs. Unattached Attributes

In a symbol drawing, you can attach attributes to symbol pins or you can add them as unattached attributes.

If the attribute applies only to a particular pin (for example, the # attribute), you should attach the attribute to the pin. If the attribute applies to the entire symbol (for example, the REFDES attribute), leave the attribute unattached.

To attach an attribute to a pin:

1. Select the pin and right-click.
2. Select the **Properties** option from the shortcut menu.
3. Select the **Attributes** tab from the **Properties** dialog box.
4. Enter the attribute name in the **Name** field and the value in the **Value** field.
5. Select the **Set** button to attach the attribute to the selected pin.
6. Click **OK** to dismiss the dialog box.

To leave an attribute unattached:

1. With your cursor placed outside of the bounding box, right double-click with nothing selected.
2. The Symbol Properties dialog box allows you to set the attributes.
3. Select the Attributes tab from the Symbol Properties dialog box.
4. Enter the attribute name in the Name field and the value in the Value field.
5. Select the Set button.
6. Click **OK** to dismiss the dialog box.

See Also

Adding Attributes

Attribute Format

Attribute Ranges

Distinguishing Labels from Attributes

Expanding Pin Attributes

Multiple and Duplicate Attributes

Multiple Attribute Values

User-Defined Attributes

Visibility Options

Attribute Format

The format for attributes is: NAME = VALUE

You must specify the attribute name exactly as you want the wirelist to interpret it. For example, to associate reference designator information with a symbol, add an attribute, such as REFDES=U?. REFDES is the attribute name and U? is the attribute value.

There are several ways to define an attribute value:

- Explicit value -- the value is constant (exactly as you enter it)

- Multiple value -- the value represents multiple attributes
- Variable value -- the value changes
- Expression value -- the value is dependent on characteristics of the design

Name and Value Restrictions

Do not create attribute names beginning with $ or *, since ViewDraw reserves names beginning with $, and since * is the wildcard character. Also avoid using % and ? in attribute values.

Do not use mathematical characters such as /,*,–, or +, in attribute values (except when specifying variable definitions for parameterized attributes).

You must enter the attribute NAME in upper case. Under certain conditions, you can enter the attribute value in upper, lower or mixed case. However, you must remember that third party tools which use the ViewDraw wirelist do not all support mixed case attribute values. These tools may require upper case values. Check with your tool provider to confirm whether you can use mixed case with the tool.

See Also

Adding Attributes

Attached vs. Unattached Attributes

Attribute Ranges

Distinguishing Labels from Attributes

Expanding Pin Attributes

Multiple and Duplicate Attributes

Multiple Attribute Values

User-Defined Attributes

Visibility Options

Visibility Options

Visibility options for attributes are:

- Visible -- makes attribute visible
- Invisible -- makes attribute invisible
- Name -- makes only the name portion of the attribute visible
- Value -- makes only the value portion of the attribute visible

You can make attributes visible or invisible at the symbol and/or schematic level. You can change attribute visibility using the Attributes tab on the Properties dialog box. If you are in a symbol window and attributes are visible, they are also visible on the schematic window. If you specify attributes visible on the symbol, you cannot make them invisible on the schematic.

Having the attributes set as visible while you are placing them on the symbol is helpful. However, as a general rule you should change the display of the attribute to invisible after you have placed them on the schematic so they do not clutter the schematic drawing.

Note: Making an attribute invisible does not delete the attribute. It only helps to reduce the clutter from the schematic display.

The # and REFDES attribute values are automatically visible on the schematic regardless of symbol level display settings. ViewDraw promotes these attributes because their values change from instance to instance, where all other attributes remain the same for each component instance.

Note: If you want the # attribute to be invisible on the schematic, as with discrete resistors, use the PINOFF attribute on the symbol. If you want the REFDES attribute to be invisible on the schematic, deselect the Display REFDES option on the Components Tab of the Project Settings dialog box.

See Also

Adding Attributes

Attached vs. Unattached Attributes

Attribute Format

Attribute Ranges

Distinguishing Labels from Attributes

Expanding Pin Attributes

Multiple and Duplicate Attributes

Multiple Attribute Values

User-Defined Attributes

User-Defined Attributes

You can create any attribute to define unique information about your design such as optional labels or part numbers used in the design. User-defined attributes can use a maximum of 128 characters to define both the name and value fields.

ViewDraw cannot interpret user-defined attributes. The wirelist process lists the attributes in the wirelist file if you add them before the wirelist is executed.

You can interpret user-defined attributes for application specific functions using ViewBase or the ViewDraw utilities.

See Also

Adding Attributes

Attached vs. Unattached Attributes

Attribute Format

Attribute Ranges

Distinguishing Labels from Attributes

Expanding Pin Attributes

Multiple and Duplicate Attributes

Multiple Attribute Values

Visibility Options

Multiple and Duplicate Attributes

Most objects can have multiple attributes (attributes with different names) associated with them. Duplicate attributes on a single object, attributes with the same name, usually have different values.

ViewDraw allows the following objects to have multiple attributes (attributes with the different names) on a single object. Duplicate attributes (attributes with identical names) on a single object usually have different values. ViewDraw allows duplicate attributes as outlined below:

Object	Duplicate
Symbol	Unrestricted duplicates
Schematics	No duplicates
Components	No duplicates (except SIGNAL,PROBE, EQUATE, and PINSWAP attributes)*
Pins	No duplicates
Net/bus segments	No duplicates (except PROBE attribute)*

* Duplicate attributes must be enabled in the configuration file if you wish to run PCB Forward. Refer to the PCB Forward Help for more information.

Most symbol attributes are not accessible from the component level. You can add identical attributes to the component and change the attribute value on the component to override the attribute value on the symbol.

See Also

Adding Attributes

Attached vs. Unattached Attributes

Attribute Format

Attribute Ranges

Distinguishing Labels from Attributes

Expanding Pin Attributes

Multiple Attribute Values

User-Defined Attributes

Visibility Options

Multiple Attribute Values

The format for multiple attribute values is:

name = *value,value,value*

Separate multiple attribute values with commas or colons. For example, #=[a,b,c,d] or TPHL=10:17:25.

The attribute TPHL=10:17:25 directs ViewSim to use one of the values as the propagation delay from high to low, depending on the specified delay. You can type either minimum (10ns), typical (17ns), or maximum (25ns).

See Also

Adding Attributes

Attached vs. Unattached Attributes

Attribute Format

Attribute Ranges

Distinguishing Labels from Attributes

Expanding Pin Attributes

Multiple and Duplicate Attributes

User-Defined Attributes

Visibility Options

Attribute Ranges

An attribute value can express a numerical range for a bus or a pin by specifying beginning, ending, and incremental integer values. ViewDraw expands an attribute value with a range to create a series of attributes.

Attribute Value Range Format:

example: The format for attribute value ranges is: NAME[F:L:I]

where F is the first number, L is the last number, and I is the interval between the numbers in the range. If no interval is specified, the interval is 1.

Separate the numbers with colons. You must enclose the range in brackets, and you can only express the range in numeric characteristics.

For

This Range...	Becomes Attributes...
#=[1:4]	#=1, #=2, #=3, and #=4
#=[1:20:2]	#=1, #=3, #=5, #=7, #=9, #=11, #=13, #=15, #=17, #=19
#=A[1:3],#=B[1:3],#=C[1:3]	#=A1, #=A2, #=A3, #=B1, #=B2, #=B3, #=C1, #=C2, and #=C3

Example

In PCB design, multiple values of the attribute #=1,4,8,12, when attached to a symbol pin and used in conjunction with a PARTS=4 attribute, indicate that this pin represents:

- pin 1 of the first part of the package
- pin 4 of the second part of the package
- pin 8 of the third part of the package
- pin 12 of the fourth part of the package

See Also

Adding Attributes

Attached vs. Unattached Attributes

Attribute Format

Distinguishing Labels from Attributes

Expanding Pin Attributes

Multiple and Duplicate Attributes

Multiple Attribute Values

User-Defined Attributes

Visibility Options

Wirelist Interpretation

ViewDraw has a flexible attributing scheme that allows an unrestricted number of attribute types (names). Wirelisting interprets almost all attributes, with each wirelist looking for a particular set of attributes.

Either flattened or hierarchical, wirelists interpret explicit and multiple attribute values. Only flattened wirelists interpret variable and expression attribute values.

Printing

Printing Documents

Before you can print in ViewDraw for the first time, you must connect the printer to the computer or network, install a printer driver, and select the printer to use. If you have not completed these tasks, please do so before attempting to print your schematic or symbol.

You can print the active window or a specified schematic sheet using the Window and Sheet options in the Print dialog box. You can also print an entire design using the File -- Print option.

Because colors that are easy to work with during a ViewDraw session are not always the best colors to use for printing, ViewDraw allows you to define a viewing color and a printing color for each object. Refer to Changing Viewing and Printing Colors for information about defining colors for graphical objects, components, text, and annotation objects.

See Also

Project Printing

Changing Viewing and Printing Colors

Printing on the PC

Printing the Current Sheet

1. Choose File -- Print.
2. Fill in the appropriate fields in the Print dialog box.

For more information about Print dialog box options, click the Help button on the dialog box.

Alternative: If you want to use default print settings, click the 🖨 button.

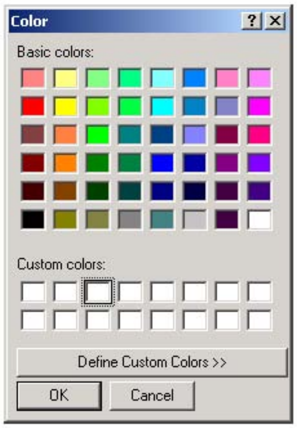
Color Options

Customizing the Color Palette

Customizing the color palette allows you to define the 16 colors to use for your screen colors and another 16 colors for project print colors.

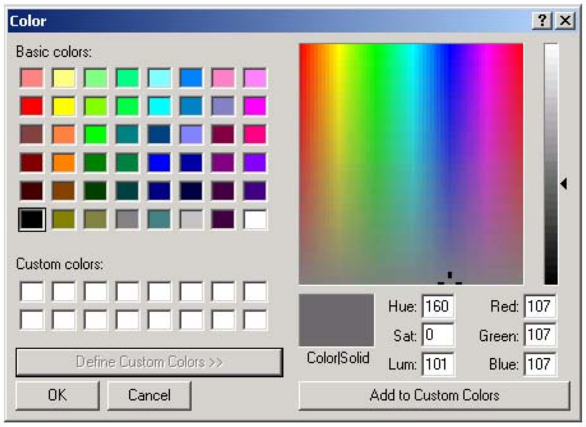
To create your own customized color:

1. Choose Project -- Settings.
2. Choose the Color Palette tab.
3. In the Choose Color field, choose the color you want to replace with the custom color.
4. Double-click the color in the color box or choose the **Edit** button.



ViewDraw Color Dialog Box

5. Select a color to edit by clicking on that color in the **Basic Colors** or **Custom Colors** box.
6. Choose the **Define Custom Colors**... button.



Hue: 160
Sat: 0
Lum: 101
Red: 107
Green: 107
Blue: 107

Defining Custom Colors

7. Change the Luminosity of your color by clicking the mouse inside the luminosity strip, dragging the cursor to affect color brightness. The LUM box indicates the new luminosity value; the defined color box displays the new color.
8. Change the color qualities: Hue, Saturation, Red, Green, and Blue. Do this by dragging the cross-hairs in the color variation box. As you move the mouse, the values for each of the qualities change depending on the position of the cross-hairs. The color defined by your choices is displayed in the color box.
9. Choose **Add to Custom Colors** to add the color to your custom color option.
10. Choose **OK** to dismiss the **Color Palette** dialog box.
11. Click **OK** to accept the new color palette and dismiss the **Project Settings** dialog box.

See Also

Changing Object Colors

Changing Viewing and Printing Colors

Exporting Color Schemes

Importing Color Schemes

Changing Object Colors

You can change how objects are displayed on your screen using the Settings command from the Project menu.

To change the color settings for a graphical object

1. Choose Project -- Settings.
2. Choose the Object Properties tab.
3. In the Object field, choose the object type whose color you want to change.
4. In the Color field, choose the color for the object.
5. In the Fill Style field, choose the fill style for the object.
6. In the Line Style field, choose the line style for the object.
7. Click OK.

Note: If the object is a text object, you can change the only the color and font style for the object.

See Also

Changing Viewing and Printing Colors

Customizing the Color Palette

Exporting Color Schemes

Importing Color Schemes

Changing Viewing and Printing Colors

ViewDraw has a default color scheme for you to use. If you have a preferred color scheme, you can change the colors ViewDraw uses. You can alter the colors of the default color scheme, or you can change the color scheme altogether.

Note: The background color for your schematic defaults to white. You can change the background color by changing the screen color for white.

To alter the colors in the default color scheme:

1. Choose Project - Settings.
2. Choose the Color Palette tab.
3. Choose the color you want to change from the Choose Color field. This list includes all the colors that are available to you. The last color in the list is your windows default background color.
4. To change the screen color for the color you selected, click the Edit button to the right of the Screen Color option. Select the color want and click OK. This changes the color you see on your screen for all objects and text mapped to the color in the Choose Color field.
5. To change the printer color for the color you selected, click the Edit button to the right of the Printed Color option. Select the color want and click **OK**. This changes the color that prints for all objects and text mapped to the color in the Choose Color field.
6. Click **OK**.

Keep in mind that non-color printers map colors to gray scale. In some cases, printers map colors to white so that you cannot see them. You can eliminate problems with color mappings by selecting the **Map all print colors to black** option.

To change the color scheme altogether:

1. Choose Project -- Settings.
2. Choose the Color Palette tab.
3. Choose a different color scheme from the Color Scheme list.
4. Click OK.

See Also

Changing Object Colors

Customizing the Color Palette

Exporting Color Schemes

Importing Color Schemes

Importing Color Schemes

You can import color schemes that have been exported from a ViewDraw session. Color scheme files have a .scm extension.

To import a color scheme:

1. Choose Project - Settings.
2. Choose the Color Palette tab.
3. Click the Import Scheme button.
4. Type the color scheme filename in the File Name field or browse to locate the file.
5. Click OK.

See Also

Changing Object Colors

Changing Viewing and Printing Colors

Customizing the Color Palette

Exporting Color Schemes

Exporting Color Schemes

You can export color schemes that you have created during a ViewDraw session for use in another project.

To export a color scheme:

1. Choose Project - Settings.
2. Choose the Color Palette tab.
3. Enter a name for the scheme in the Color Scheme field.
4. Click the **Export Scheme** button.
5. Type a file name for the color scheme in the **File Name** field.
6. Click **OK**.

Note: You can concatenate color scheme (.SCM) files together to keep a master list of all schemes. When imported, the new schemes appear in the drop down Color Scheme list.

See Also

Changing Object Colors

Changing Viewing and Printing Colors

Customizing the Color Palette

Importing Color Schemes

Scaling Fonts and Components

Scaling Fonts and Components

ViewDraw allows you to scale components or fonts.

To scale components or a selected instance of a font:

1. Select the component or font.
2. Click the button.
3. Enter the Scale Factor you want to use.
4. Click **OK**.

To scale fonts:

1. Select Project -- Settings.
2. Select the Font Tab.
3. Enter the Scale Factor you want to use.
4. Click OK.

See Also

Edit -- Scale command

Project -- Settings command

Linking and Embedding Objects

Inserting Objects

You can include information or documents created in other applications in your ViewDraw document. You can link or embed documents or objects in your ViewDraw documents. The main difference between linking and embedding is where the data is stored. If you embed the object, the data becomes part of your ViewDraw document. If you link the object, the ViewDraw document stores only the location of the information and then displays a graphic representation of the information in the ViewDraw document.

See Also

Changing Object Colors

Converting an Embedded Object to a Different File Format

Deselecting Objects

Embedding an Object

Linking Objects

Paste Special dialog box

Selecting Objects

Embedding an Object

You can embed a new or existing object into your ViewDraw document.

To embed a new object:

1. Position the insertion point where you want to embed the object in the document.
2. Choose **Edit - Insert Object**.
3. Select the **Create New** option.
4. Select the **Object Type** that you want to create.

   The list of options in this list depends on the applications you have installed on your computer. You cannot insert objects created by an application that you do not have installed.
5. Click **OK**.
6. Return to ViewDraw.
7. Exit the application used to create the object or click anywhere in the ViewDraw window.

To embed an existing object:

1. Position the insertion point where you want to embed the object in the document.
2. Choose **Edit - Insert Object**.
3. Select the **Create from File** option.
4. Enter or select the file name of the object you want to embed.

   You can use the **Browse** button to search for and select the file.
5. Click **OK**.
6. If you want to display the object as an icon, click **Display** as Icon check box before clicking OK.
7. Return to ViewDraw.
8. Click anywhere in the ViewDraw window.

See Also

Changing Object Colors

Converting an Embedded Object to a Different File Format

Deselecting Objects

Inserting Objects

Linking Objects

Paste Special dialog box

Selecting Objects

Converting an Embedded Object to a Different File Format

You can convert the object to a different file format by selecting a different application to be the object's source application.

To convert an embedded object to a different file format:

1. Select the embedded object whose source application you want to change.
2. Choose the name of the object you want to convert and then choose **Convert** from the **Edit** menu.
3. Do one of the following:

   To permanently convert the object, select the Convert To button.

   To temporarily activate all embedded objects if the selected type in the file format you specify in the Object Type box, select the Activate As option button.

When you edit the object, you use the file format you specified, but when you save the object, it is saved in its original file format.

4. In the **Object Type** box, select the application whose file format you want to convert the embedded object to.
5. Click **OK**.

See Also

Changing Object Colors

Deselecting Objects

Embedding an Object

Inserting Objects

Linking Objects

Paste Special dialog box

Selecting Objects

Linking Objects

You can create a link to an object to share the object between two ViewDraw documents or a ViewDraw document and a file created in another application. You must be running both applications, and both applications must support dynamic data exchange (DDE) or object linking and embedding (OLE).

To create a link to another file or Word document:

1. Make sure that you save the source file before you link the information.
2. In the application in which the information you want to link was created, open the source file and then select the information you want to link.
3. Choose **Edit - Copy**.
4. Switch to the ViewDraw document and position the cursor on the insertion point where you want to insert the linked information.
5. Choose **Edit - Paste Special**.
6. Select the Paste Link Option button. In the As box, select the appropriate option.
7. Click **OK**.

See Also

Breaking a Link

Changing Object Colors

Converting an Embedded Object to a Different File Format

Deselecting Objects

Editing Linked Information

Embedding an Object

Inserting Objects

Paste Special dialog box

Updating a Link

Selecting Objects

Updating a Link

ViewDraw can update linked information in a ViewDraw document whenever the information in the source file changes. You can specify automatic or manual updating for each link. ViewDraw automatically updates links when you open the ViewDraw document and any time the source file changes while the ViewDraw document is open. ViewDraw updates manual links only when you choose the Update Now button on the Links dialog box.

To define update rules for links:

1. Choose Edit - Links.
2. Select the link you want to define the update rules for. To select multiple links, hold down Ctrl while you click each link.
3. Select Automatic or Manual to define the update rule.
4. Click OK.

To update links manually:

1. Choose Edit - Links.
2. Select the link you want to update.
3. Choose the Update Now button.
4. Choose the Close button.

See Also

Breaking a Link

Changing Object Colors

Converting an Embedded Object to a Different File Format

Deselecting Objects

Editing Linked Information

Embedding an Object

Inserting Objects

Linking Objects

Paste Special dialog box

Selecting Objects

Breaking a Link

You can break a link to a source file if you no longer need updated information from the source file.

To break a link:

1. In the ViewDraw destination document, choose **Edit - Links**.
2. Select the link or links you want to break.
3. Select the **Break Link** button.
4. Click **Close**.

See Also

Changing Object Colors

Converting an Embedded Object to a Different File Format

Deselecting Objects

Editing Linked Information

Embedding an Object

Inserting Objects

Linking Objects

Paste Special dialog box

Updating a Link

Selecting Objects

Editing Linked Information

Always edit linked information in the source file so the changed information is reflected in both the source and destination documents when you update the links.

To edit linked information:

1. Select the linked information to edit.
2. Choose the name of the link and then choose **Edit - Open Source**.
3. Make your changes in the source file and then save the changes.

See Also

Breaking a Link

Linking Objects

Updating a Link

Manipulating Objects

Selecting Objects

You select objects for a variety of reasons:

- to edit the properties of the object
- to move or relocate the object
- to zoom in on a specific object

When you select an object, the object outline is highlighted. If a selection results in a bounding box surrounding the objects, it means that you have selected a component that contains more than one object or text string.

When you select a net, bus, pin, or component, all associated labels and attributes are also selected.

The select function now hunts for the correct item to select. If you have the mouse pointer over a group of objects, the Select function cycles through all the objects in the selection distance one at a time, advancing to the next object each time you press F2 (or select again).

To select individual objects:

1. If you are not in select mode, click the ↖ button or press the Esc key to enter select mode.
2. Place the cursor on the object you want to select and click it.

All previously selected objects are deselected.

-Or-

1. Click and drag the cursor to specify the selection area.
   ViewDraw selects only objects completely within the selection box.

Note: If the item you want is in the selection area but not selected, press F2 (or select again) to cycle through the objects in the selection distance until the correct object is selected.

To select all of the same objects:

1. Choose Edit - Select to use the **Select** dialog box to select an object. The **Select** dialog box allows you to specify the selection criteria.
2. Select the object type to select from the **Object Type** list.
3. Enter the symbol, attribute, label, name, or text string in the **Expression** field.
4. Click **OK** to select the specified object(s).

To select a net or bus:

1. If you are not in the select mode, click the ↖ button or press the Esc key to enter select mode.
2. Place the cursor on a segment of the net or bus and click it.
3. With the cursor on the selected segment, press **Ctrl** and click the mouse again.

ViewDraw selects the entire net or bus up to the solder joint or a pin.

Note: If the item you want is in the selection area but not selected, press F2 (or select again) to cycle through the objects in the selection distance until the correct object is selected.

To select components:

1. If you are not in select mode, click on the ↖ button or press the Esc key to enter select mode.
2. Place the cursor on the component and click it.

Selecting the component using the mouse selects only the component you clicked. If you want to select all symbols with the same name, use the **Select** command from the Edit menu.

To select multiple components with the same name:

1. Choose Edit - Select to use the Select dialog box to select the component.
2. Choose **Component** from the **Select** list.
3. Enter the symbol name in the **Expression** field.
4. Click **OK**.

The **Select** command selects all components in the schematic with the symbol name you specify.

To select multiple components or objects:

1. If you are not in select mode, click on the ↖ button or press the Esc key to enter select mode.
2. Click the first component or object to select it.
3. Press the **Ctrl** key while you click and drag to select a group of objects, press the **Ctrl** key while click and drag over the group of objects you want to select.

-Or-

1. Choose Edit - Select from the Edit menu to use the Select dialog box to select the each component or object. Choose the appropriate object type from the Select list.
2. Enter the appropriate name, text, label, or attribute in the Expression field.
3. Click **Add Select**.
4. Repeat these steps for each object you want selected.

Note: When selecting a net segments, if you click twice on a net segment while holding the Ctrl key, the entire net is selected. The first click selects the individual net segment, the second click selects the entire net.

See Also

Changing Object Colors

Converting an Embedded Object to a Different File Format

Creating a New Schematic

Creating a New Symbol

Creating a Pin Symbol

Creating a Sheet Border

Creating an Annotate Symbol

Creating Composite Symbols

Deselecting Objects

Embedding an Object

Inserting Objects

Linking Objects

Paste Special dialog box

View -- Full command

View -- In command

View -- Out command

View -- Pan command

View -- Restore Zoom command

View -- Save Zoom command

View -- Selected command

View -- Zoom command

Deselecting Objects

A selected object remains selected until you deselect it, delete it, or select something else.

To deselect all selected objects:

Place the cursor in an area of the window with no objects and click the mouse button.

To deselect a specific object:

Position the cursor on the object and press **Shift** and click the mouse button.

To deselect nets, net segments, and buses:

1. Position the cursor on the segment you want to deselect.
2. Press **Shift** and click the left mouse button.

If you specify a selected segment for deselecting, only that segment is deselected. If you specify a deselected segment, the entire net or bus is deselected.

To deselect a group of objects:

1. Press the Shift key.
2. Click and drag to form a box over the group or area that you want to deselect.
3. Release the mouse button.
   ViewDraw deselects all selected objects located completely within the box. If any part of an object appears outside of the box, ViewDraw does not deselect the object.

See Also

Adding Components

Changing Object Colors

Converting an Embedded Object to a Different File Format

Embedding an Object

Inserting Objects

Linking Objects Paste Special dialog box

Selecting Objects

Moving Objects and Components

You move objects by selecting and dragging the objects and components in the active window. You can move specific components, such as pins in a symbol window, and complete components in the schematic window.

To move an object:

1. Click and drag the object to select it.
2. While still holding the mouse button, drag the object to the new location.

If you release the mouse button and click it again, ViewDraw displays the properties dialog box for the object or component you selected. If this happens, close the dialog box, deselect the object, and try this procedure again.

To move a group of objects:

Select the objects you want to move. Drag the selected objects to the new location.

See Also

Changing Object Colors

Converting an Embedded Object to a Different File Format

Deselecting Objects

Embedding an Object

Inserting Objects

Linking Objects

Paste Special dialog box

Selecting Objects

Edit -- Stretch command

Stretches the selected object in any direction. Stretchable objects are: Lines, Boxes, Circles, Arcs, and Pins.

To stretch an object:

1. Select the object or group of objects you want to stretch.
2. Choose **Edit - Stretch** or click on the toolbar.
3. With the left mouse button, use the cursor to grab the object and drag it to the shape and size you want.
4. Release the mouse button.

Shortcut

Toolbar:

Command Line: stretch

Edit -- Delete command

Deletes the selected objects in the active document. The delete command can be reversed using the Undo command.

If you delete a component, net segment, or bus segment from a schematic, all connections become dangling connections.

If the object you delete has dangling connections, by default these wires (including attached labels and attributes) are also deleted. If you want to preserve net labels and attributes from deletion in this case, use **Ctrl+ del**.

Shortcut

Toolbar: ✏

Key Bindings:

PC: **Delete** or **d**

Command Line: **delete**

Edit -- Copy command

Copies the selected objects or areas of the drawing into the clipboard, overwriting the previous contents of the clipboard.

To copy a selected object:

1. Select the object or group of objects you want to copy.
2. Choose **Edit - Copy**.

To paste the object:

1. Choose Edit - Paste.
2. Click and hold the left mouse button to determine where you want the object pasted.
3. Once you have the object in place, release the mouse button.

Shortcut

Toolbar: ⧉

Key Bindings:

PC: **Ctrl+C**

Command Line: bcopy

Edit -- Cut command

Removes the selected objects or areas of the drawing and places them on the clipboard, overwriting the previous clipboard contents. Use this command when you want to paste the objects elsewhere in another type of document.

To cut a selected object:

1. Select the object or group of objects you want to cut.
2. Choose **Edit - Cut**.

Shortcuts

Toolbar: ✂

Key Bindings:

PC: **Ctrl+X**

Command Line: bcut

Edit -- Paste command

Pastes the contents of the clipboard into the drawing at the location you specify. Use this command when the clipboard contains something you want to paste in your schematic.

To paste the object

1. Choose Edit - Paste.
2. Click and hold the left mouse button to determine where you want the object pasted.
3. Once you have the object in place, release the mouse button.

Note: When performing a paste, start with the cursor in the approximate location you want the lower left corner of the paste to land. Press and hold the left mouse button as you move the mouse slightly. This causes the paste to appear on the cursor and you can then place it as necessary prior to releasing the left mouse button. You can use the function keys while dragging the pasted objects to move around the schematic.

Shortcuts

Toolbar: 📋

Key Bindings:

PC: **Ctrl+V**

Command Line: **bpaste**

Creating Analog Symbols

Creating the Underlying Schematic for an Analog Symbol

Before you create the underlying schematic, you must have the Analog library defined in your project. If you do not have this set up, go to ??? to do this before you go on.

To create the underlying schematic for the symbol created in Creating Analog Symbols:

1. Choose **File - Open**.
2. Select Schematic as the type and enter a name in the Schematic text box.
   A symbol and its underlying schematic must have the same name. This is the only way the system knows that the two are associated.
3. Click **OK** to create the schematic window.
4. Add the N-type MOSFET component using the Component command from the Add menu.
   The N-type MOSFET component is in the analog library and it is named NFET.1.
5. Add the P-type MOSFET component using the Component command from the Add menu.
   The P-type MOSFET component is in the analog library and it is named PFET.1.
6. Select the P-type MOSFET and choose the Edit - Rotate command to rotate the P-type MOSFET. Rotate the component until the emitter is away from the schematic center.
7. Add the ground symbol.
8. Draw the nets.
   For more information on adding nets, refer to Adding Nets and Buses for Connectivity.
9. Create labels to name three signals. Perform this step for VA, VIN, and OUT.

The names assigned to the underlying schematic nets must be the same as the names assigned to the symbol pins. For more information on adding labels, refer to Adding Labels.

10. Choose **File - Save** to save the Schematic.

See Also

Creating Analog Symbols

Transformers

The Analog parts library contains two transformers symbols; TRANSF and TRANSFCT. TRANSF is an abbreviation for Transformer and TRANSFCT is an abbreviation for Transformer, Center-Tapped.

Type of Symbols

The TRANSF symbol has both an underlying schematic and a model file, tranfs.mod.

If a symbol is specified as a composite block type, the MODEL attribute is ignored and the underlying schematic and any attributes assigned to this level are the reference source file for default parameters. All customization for composite symbols must be done in the underlying schematic.

A symbol specified as a module block type relies on the MODEL attribute to point to the .mod file, where default parameters are defined. To customize a module symbol, you need to edit the .mod file. This is a sample mod file:

```
* MODEL FILE FOR "TRANSF" TRANSFORMER SYMBOL
* To change the parameters of the transformer,
* copy this model file to another file and perform
* the following steps:
*
* 1) Change the name of the subckt from
* "TRANSF" to the same name as your file.
* 2) Change the value of the primary inductance.
* 3) Change the value of the secondary inductance.
* 4) Change the coefficient of coupling.
* 5) Change the values of the primary and
* secondary DC resistance.
*
* These parameters are underlined.
*
.SUBCKT TRANSF 1 2 3 4
*
LPRIM 5 2 120MH
*
RPS 1 5 5.0
*
LSEC 6 4 20MH
*
RSS 3 6 .001
*
KPS LPRIM LSEC .999
*
.ENDS
```

See Also

Creating the Transformer Schematic

Creating a Transformer Symbol

Using the Ampersand (&) Token

Creating a Transformer Symbol

You can use the supplied transformers, or you can create your own transformers for specific applications. The Analog parts library offers a symbol called TCORE that you can use as the core for any transformer you create. The cores and inductors in this symbol file are only graphics created at the symbol level using the Add menu commands.

To create a transformer:

1. Choose File - Open.
2. Select Symbol from the Type and enter a name in the Symbol field.
3. Create the symbol using the Arc, Box, Circle, and Line commands from the Add menu. Keep in mind that you must use pins to connect represented circuitry to the bounding box, otherwise you will have a symbol that cannot be connected to anything on an upper level schematic.
4. Choose File - Save to save the symbol.

See Also

Creating the Transformer Schematic

Transformers

Using the Ampersand (&) Token

Creating the Transformer Schematic

You must create an underlying schematic for the transformer symbol.

1. Choose **File - Open**.
2. Select Schematic from the Type and enter the schematic name in the field.
3. Name the schematic the same as the transformer symbol.
4. Add required vertical inductors. Choose the **Component** command from the **Add** menu (or click D) and select the **ind.2** component from the Analog library.
5. Add the transformer core to the schematic. Choose the **Component** command from the Add menu (or click D) and select the tcore component from the Analog library.
6. Add required nets. Choose the Nets command from the **Add** menu (or click ⊣ to add a net to the schematic.

For more information about adding nets, refer to Adding Nets. Adding Nets and Buses for Connectivity

Note: The net labels in the schematic must be the same as the pin labels in the symbol and there must be a corresponding labeled net for every pin on the symbol. There may be additional labels on the schematic that do not correspond to the symbol, but they cannot be connected through pins.

7. Follow these steps to modify the ORDER attribute of the transformer core:
   1. Select the Transformer core and double-click to select the **Properties** option.

2. Choose the Attribute tab of the **Properties** dialog box.
3. Change the ORDER attribute's value to "A& B& C& D& COEFFICIENT$".
4. Add a letter and the & sign for each inductor. Add the COEFFICIENT$ value at the end of the list. Refer to Using the Ampersand `&" Token for more information.
5. Choose the **Set** button.
6. Select **OK** to dismiss the dialog box.
7. Select each of the inductors and make note of their unique IDs. The unique ID appears in the lower left corner of the schematic window and looks similar to:$1I2 or $1I8.
8. Select the transformer core and double-click to select the Properties option.
9. Choose the Attribute tab of the Properties dialog box.
10. Add new attributes for each inductor and for the COEFFICIENT value.
    A = unique_id
    B = unique_id
    C = unique_id
    D = unique_id
    COEFFICIENT = .999

    When assigning unique_id values to attributes DO NOT use hierarchical unique Ids such as $1I2\$1I3\$1I4. Use only the ID after the last backslash, such as $1I4.

    Do not create the new attributes on the inductors themselves.
11. Choose the **Set** button.
12. Select **OK** to dismiss the dialog box.
13. Assign component values to each inductor. This can be done with a Spice dialog box. Use the following values:
    Upper left: 300U Upper right: 200U
    Lower left: 200U Lower right: 200U

8. Choose **Save** from the File menu to save the schematic.

You now have a custom transformer, which can be used with a hierarchical design, consisting of a symbol for use on an upper-level schematic and an underlying schematic.

See Also

Creating a Transformer Symbol

Transformers

Using the Ampersand (&) Token

Using the Ampersand (&) Token

The '&' token is used in the ORDER attribute. It automates the coupling of components and sorts out component instances in the wirelist. It is particularly useful when copying components.

Using the Creating the Transformer Schematic procedure as an example, the A& in the core ORDER attribute tells the system that the core (K1) has an attribute called 'A' which has a unique ID assigned to it. This unique ID acts as pointer to another component. If A=$1I19, this points to the l? instance with that ID.

The L? instance with $1I19 ID on the schematic is described by the L19 line in the wirelist. The K1 line references the L19 line. Look at the L19 line in the following sample wire file and notice that it shows C1, TAP, and 300U. This means that in the underlying schematic, the 300Uh inductor is between the CI and TAP and has a unique ID of $1I19. The L19 designation in the wirelist is derived from the component's prefix and unique ID. This is how the '&' token keeps track of each component.

The K1 line in the wirelist file shows that K1 mutually couples the four inductors together using a coefficient of .99. Here the ‘&’ token keeps the component designations the same. If you had used the ‘$’ token in the ORDER attribute, this line would show the unique IDs of each component, making it more difficult to relate the components on the K1 line to the specific component description lines above it.

Sample Wirelist File:

```
* Project Token
* Wirelist Created with Version X.XX


L19 C1 TAP 300U


L20 S1P 0 200U


R2 0 S2P 10K


R1 S1P 0 5K


L4 TAP 0 200U


R3 N40 TAP 50S


IIN 0 C1 AC 1 0


V50 N40 0 DC 15V


L6 0 S2P 200U


K1 L19 L4 L6 L20


.PROBE/CSDF


.AC DEC 10 1KHZ 1GHZ


* DICTIONARY 2
* $1N40 = N40
* GND = 0
.PRINT AC V([C1]) V([TAP]) V([S1P]) V([S2P])
.END
```

See Also

Creating the Transformer Schematic

Creating a Transformer Symbol

Transformers

Using FUBs

What is a Fub?

A fub is a symbol shell or a symbol under construction. It is the framework for a symbol, which you build on to make the actual symbol. Using the fub functionality in ViewDraw, you get the following:

- Automatic pin updates when you add nets or buses to the fub:

When you add nets or buses to the fub, ViewDraw immediately updates the fub to include the pins.

- Automatic pin labeling when you save the symbol.

When you save the symbol using the Save command from the File menu, ViewDraw labels the pins so that they correspond to the nets or buses they connect to. Automatic pin labeling does not happen until you save the symbol.

You cannot copy or duplicate a fub. If you try to do this, you will get an error message.

Before you convert the fub into the symbol, you must remove the SYMTYPE=FUB attribute from the symbol. This attribute activates all the fub functionality. When you remove this attribute, the fub is converted to a symbol.

Once you make the fub into a symbol by removing the SYMTYPE=FUB attribute and by adding other appropriate attributes, you can then copy or duplicate the symbol. For more information on:

- creating a symbol using fubs, refer to Adding a Fub.
- adding and removing attributes, refer to Adding Attributes.

Identifying a FUB

FUBs are essentially instantiations of symbols that have a special property that allow you to manipulate them at the schematic level. Unless otherwise specified, you can edit FUB symbols just like non-FUB symbols by pushing into the FUB symbol definition, and instances.

You use the SYMTYPE=FUB attribute to distinguish a FUB symbol from a non-FUB symbol. When you create a FUB using the ADD -- Fub command, ViewDraw automatically creates the SYMTYPE=FUB attribute on the FUB symbol definition. By default, this attribute is invisible. The Project -- Block -> Detail command indicates which components are FUBs by specifying (FUB) next to the component listing.

See Also

Adding a Fub

Adding and Deleting the SYMTYPE=FUB Attribute

FUB Graphics

FUB Names

FUB Pin Creation

FUB Pin Deletion

FUB Restrictions

Saving FUBs

Saving FUBs

FUB symbols are written to disk whenever the schematic that contains an instantiation of the FUB is written to disk, namely by the File - Save, File - Save Copy As, and File - Save and Check commands. You can push into the FUB symbol and save it at any time.

FUB symbols are written to log files at the same time that the schematic is written to a log file.

See Also

Adding a Fub

Adding and Deleting the SYMTYPE=FUB Attribute

FUB Graphics

FUB Names

FUB Pin Creation

FUB Pin Deletion

FUB Restrictions

Identifying a FUB

FUB Pin Creation

When creating or moving a net or a bus with the **Add - Net** or **Add - Bus** commands, if an endpoint on the net or bus falls on the bounding box of a FUB, ViewDraw automatically creates and connects a FUB pin to the net or bus. The FUB pin extends from the outer FUB box to the inner FUB box.

Note: The creation of the FUB pin clears the schematic's Undo and Redo stacks, therefore disabling the operation of these commands.

ViewDraw determines the value of the PINTYPE attribute from the side of the symbol where you create the pin.

If you create a symbol pin on the…	…Then ViewDraw assigns the following attribute value…
left side	PINTYPE=IN
right side	PINTYPE=OUT
top or bottom	PINTYPE=BI

See Also

Adding a Fub

Adding and Deleting the SYMTYPE=FUB Attribute

FUB Graphics

FUB Names

FUB Pin Deletion

FUB Restrictions

Identifying a FUB

Saving FUBs

FUB Pin Deletion

When deleting a net or bus segment using the **Edit - Delete** command, if the segment is attached to a FUB pin, ViewDraw automatically deletes the FUB pin.

Note: The deletion of the FUB pin clears the schematic's Undo and Redo stacks, therefore disabling the operation of these commands.

See Also

Adding a Fub

Adding and Deleting the SYMTYPE=FUB Attribute

FUB Graphics

FUB Names

FUB Pin Creation

FUB Restrictions

Identifying a FUB

Saving FUBs

FUB Pin Labels

FUB pins get their names from the attached nets. During a schematic File - Save, File - Save Copy As, and File - Save and Check command, ViewDraw assigns labels to the FUB pins. The value of these labels is the names of the attached net or bus. If you did not label the attached net or bus, then ViewDraw assigns the internal name of the net or bus. If you did label the attached net or bus, ViewDraw assigns the label string itself. If the FUB pin already has a label, ViewDraw overrides its value.

Note: If two nets with the same name are attached to the same FUB, the pins are given the same name and will become pass-through pins.

See Also

Adding a Fub

Adding and Deleting the SYMTYPE=FUB Attribute

FUB Graphics

FUB Names

FUB Pin Creation

FUB Pin Deletion

FUB Restrictions

Identifying a FUB

Saving FUBs

Adding and Deleting the SYMTYPE=FUB Attribute

You may manually add and delete the SYMTYPE=FUB attribute from any symbol. If you delete the attribute from a FUB, the symbol will no longer behave as a FUB, and you will no longer be able to modify the symbol at the schematic level. If you add the SYMTYPE=FUB attribute to a symbol, them it will behave as a FUB.

Note: ViewDraw expects FUBs to graphically contain two boxes with a static pin size. Your pins may look awkward if you try to turn just any symbol into a FUB. In addition, the Tools -- Check Project command will generate an error if it finds the same FUB instantiated more than once on a single page.

See Also

Adding a Fub

FUB Graphics

FUB Names

FUB Pin Creation

FUB Pin Deletion

FUB Pin Labels

FUB Restrictions

Identifying a FUB

Saving FUBs

Instantiating a VHDL/Verilog Model

Instantiating a Verilog Model in a ViewDraw Schematic

You can include Verilog models as symbols in a ViewDraw schematic.

To include a Verilog model:

1. Create a ViewDraw symbol that represents the Verilog or VHDL model. You create the symbols in ViewDraw. The symbol should be a Module symbol.
   Click here for information about creating symbols in ViewDraw. Make sure that the symbol type is Module or Composite. For type Composite there is an additional step. You use Composite if you are using Intelliflow or plan to replace your source code with a schematic.
2. Add the symbol as a regular symbol into a schematic.

Advanced Features

Advanced Attributes

Advanced Attributes

Advanced attributes are attributes used in ViewDraw. ViewDraw supports Parameterized Attributes

A parameterized attribute has a value field that is a variable. This variable specifies values that change in the design or change from design to design.

Wirelisting interprets parameterized attributes for the target simulator or layout system. Wirelists that generate a flattened connectivity description (hierarchy expanded and removed) can usually take full advantage of the macro expansion and expression evaluation features of parameterized attributes. The target system restricts wirelists that generate a hierarchical connectivity description to the parameter passing capabilities of the target system.

See Also

Attribute Expression Values and Operators

Attribute Variable Definition

Default Variable Definitions

Primary Variable Definitions

Project Variable File

The Expansion Process

Using Parameterized Attributes

Using Parameterized Attributes

Parameterized attributes have a value field that is a variable. This variable value specifies values that change in the design under different conditions, or that change from one design to another. An attribute value field may be an expression that is arithmetic, relational, or logical. Expressions may also contain strings (which may be concatenated) and variables that have been previously defined.

Wirelisting interprets parameterized attributes for the target simulator or layout system. Wirelists that generate a flattened connectivity description (hierarchy expanded and removed) can usually take full advantage of the macro expansion and expression evaluation features of parameterized attributes. The target system restricts wirelists that generate a hierarchical connectivity description to the parameter passing capabilities of the target system.

If you are using external simulation or layout systems, evaluate the capabilities of the external tools and the interfaces to each tool when integrating parameterized attributes into the design. The system interface documentation usually describes the support of parameterized attributes.

See Also

Advanced Attributes

Attribute Expression Values and Operators

Attribute Variable Definition

Default Variable Definitions

Primary Variable Definitions

Project Variable File

The Expansion Process

The Expansion Process

Because a variable can point to another variable, the system repeats the expansion process until all variables are evaluated.

The system follows these steps to perform the expansion process to replace variables with explicit values:

See Also

Advanced Attributes

Attribute Expression Values and Operators

Attribute Variable Definition

Default Variable Definitions

Primary Variable Definitions

Project Variable File

Using Parameterized Attributes

Attribute Expression Values and Operators

Used in the value field of an attribute, an expression is a statement that contains a minimum of two values and one or more operators that define the performance of operations on these values. The values used in an expression can be explicit values, variable values, or other expressions. During the expansion process, the system performs the operations specified in an expression to obtain an explicit value for the attribute.

Attribute Expression Operators

You can use three types of operators in expressions to define the type of operation:

- Arithmetic — For simple arithmetic operations

  ```
  * — Multiplication

  / — Division

  + — Addition

  - — Subtraction
  ```

- Boolean — For simple logical functions

  ```
  ! — Logical Not

  && — Logical And

  || — Logical Or

   — Exclusive Or (space)
  ```

- Miscellaneous — For nonlogical operations

  ```
  () — Separates operations

  "string" — Defines a text string

  # — Joins two values

  -- — Denotes negative integer
  ```

Note: Do not mix arithmetic and Boolean operators in the same expression.

Example 1

When other characteristics of your design affect or support the definition of attribute values, expressions simplify defining those attributes. For example, you can define TPHL as follows:

```
TPHL=@MAX_DELAY + @TEMP_COEF
```

because the high to low propagation delay of an object might be the sum of maximum delay and temperature coefficient. To determine the explicit value of the TPHL attribute, the system adds the explicit values assigned to the variables @MAX_DELAY and @TEMP_COEF. The variable definitions are:

```
@MAX_DELAY=6.2E-3

@TEMP_COEF=2.34-3
```

The explicit value of the TPHL attribute is:

```
TPHL=8.4E-3
```

Example 2

Define the CAP attribute as follows:

```
CAP=(1+2) * 6.1E1
```

The result is:

```
CAP=1.83E2
```

The system automatically increases the exponential factor from E1 to E2 because the integer 18.3 is written as 1.83E1 in scientific notation.

See Also

Advanced Attributes

Attribute Variable Definition

Default Variable Definitions

Primary Variable Definitions

Project Variable File

The Expansion Process

Using Parameterized Attributes

Bus Rippers

Bus Rippers

A bus ripper is a symbol that you instantiate in your design to pull signals from a bus. This object — the bus ripper symbol — provides you with an unambiguous reference to selected signals of the bus. You can also use bus rippers to create an alias net name for signals that you specify. If you do not intend to create aliased net names, you can extract signals from a bus the traditional way (without using a bus ripper).

To enable Bus Rippers

Before you can use bus rippers, you must enable the RIPPERS_ON keyword in the **viewdraw.ini** file. Set the bus rippers to on using the Project Settings command. Use the Settings command from the Project menu to check the Bus Rippers check box.

Once you save a design that has bus rippers enabled, the version of the design database changes. This means your design is no longer backward compatible and, therefore, cannot be read into applications that have not been rebuilt with the latest libraries (such as ViewBase). Therefore, if you want to use bus rippers and net aliasing, you must rebuild your applications (ASIC design kits, netlisters, delay calculators, design rule checkers) with the latest libraries.

For BIT attributes to propagate from the bus ripper symbol to the schematic level, you must ensure that the BIT attribute is an argument to the ATTR_PROMOTE keyword in the **viewdraw.ini** file.

For example: `ATTR_PROMOTE BIT`

Set the ATTR_PROMOTE keyword using the Project - Settings command. Select Promote from the Attribute List on the Attributes Tab and enter BIT in the name field (if it is not already there).

See Also

Bus Ripper Symbols

Bus Rippers and Bundles

Connecting a Bus Ripper to a Bus

Extracting Signals From a Bus

Miscellaneous Advanced Features

Net Aliasing

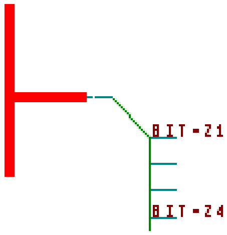
Resolving Net Names

Using Bus Rippers

Connecting a Bus Ripper to a Bus

You use the **Component** command from the Add menu to add a bus ripper to your schematic. The Builtin library contains two bus ripper symbols (4b_rip, and 8b_rip). When you instantiate a bus ripper symbol in your design, you can not connect the symbol directly to a bus by abutment (except when you connect to an endpoint of the bus). You must extend a segment from the bus and connect it to the pin that is designated as input to the bus ripper. This is the pin identified by an argument to the WHICHRIPPED attribute.

Note: For more information on port mapping, refer to Bus Ripper Symbols.



Connecting a Bus Ripper To a Bus

See Also

Bus Ripper Symbols

Bus Rippers

Bus Rippers and Bundles

Creating a New Schematic

Extracting Signals From a Bus

Net Aliasing

Resolving Net Names

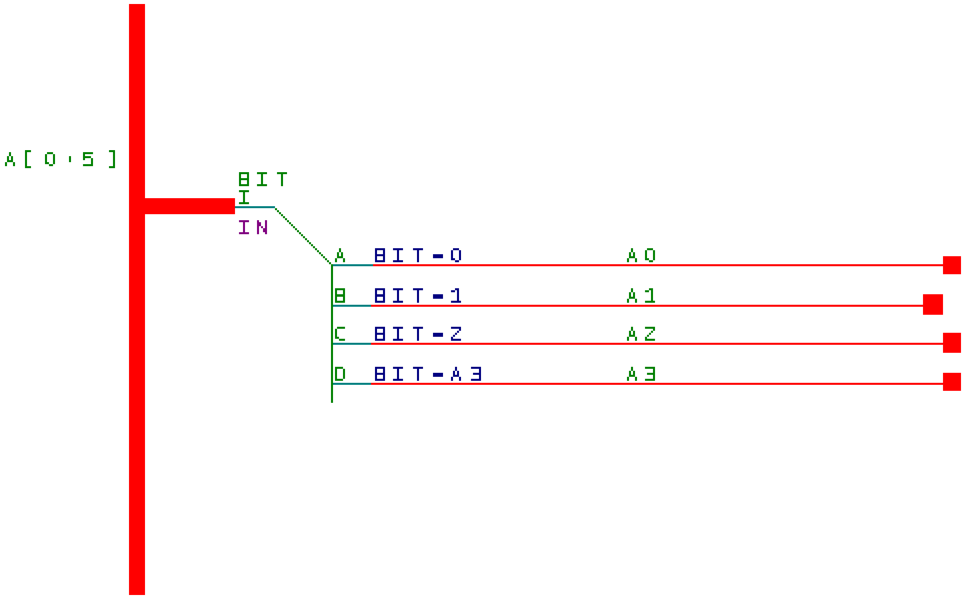
Using Bus Rippers

Extracting Signals From a Bus

You extract a signal from a bus by connecting a net to one of the pins of the bus ripper that has a BIT attribute. You identify the bus signal to extract by relabeling the value of the BIT attribute to correspond to the desired signal (`BIT=`*`net_name`* `or` `BIT=`*`position`*). To change the BIT attribute, select it and right-click to activate the Properties dialog box.

Net Name or Position

You can extract a signal from a bus by specifying the net’s name or by specifying the net’s physical position within the bus. For example, the figure below shows two signals pulled from a bus. The first net is identified by physical position within the bus (`BIT=0`); the last net is identified by name (`BIT=A3`).



Notation by Net Name or by Position

Initial Index of a Bus

The value of the BIT attribute on the component pin of a bus ripper specifies the signal to pull from a bus. You can specify the BIT attribute value as (1) the *name* of the net, or (2) the physical *position* of the net within the bus.

Differing Initial Bus Indices

The bus on the left in the Figure 1 (`A[0:5]`) has its index starting at zero; the bus on the Figure 2 (`A[1:5]`) has its index starting at one. Suppose that you specify BIT=0 and BIT=3 as component pin attributes on each bus ripper. Then the nets connected to the component pins of the bus ripper on the:

- *Figure 1* indicates implicit values of `A0` and `A3`, respectively.
- *Figure 2* indicates implicit values of `A1` and `A4`, respectively.

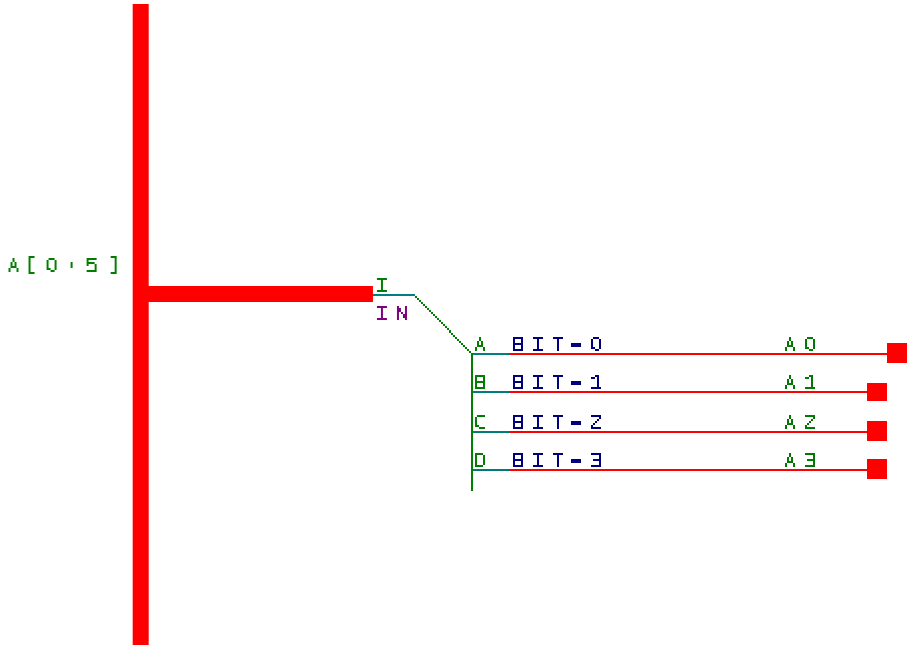


Figure 1: Bus Index Starting at Zero

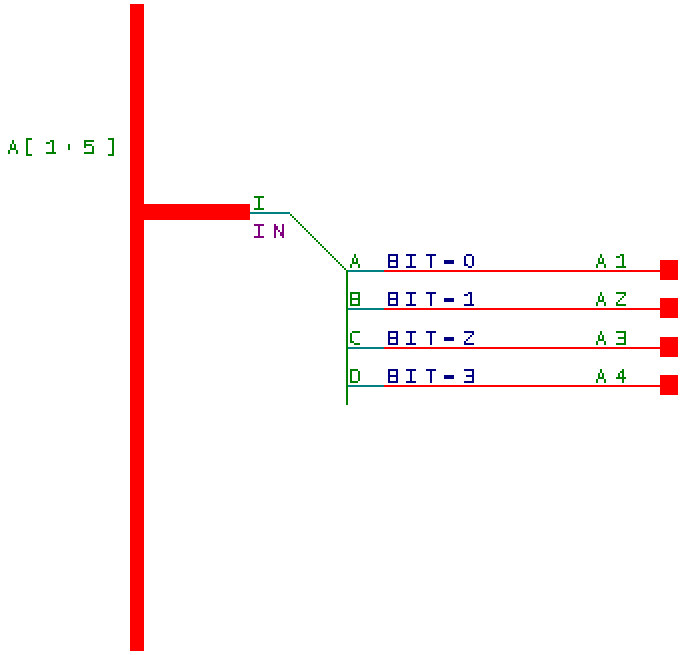


Figure 2: Bus Index Starting at 1

Figure 3 and Figure 4 show two buses. In Figure 3, a 6–bit bus (`A[0:5]`) starts at zero; in Figure 4, a 5–bit bus (`A[1:5]`) starts at one. Each bus has 4 signals pulled from it by a bus ripper. Notice that using the same BIT attributes on each component pin yields different signals. Also notice that the bottom two nets of the bus ripper on Figure 4 pull the same signal.

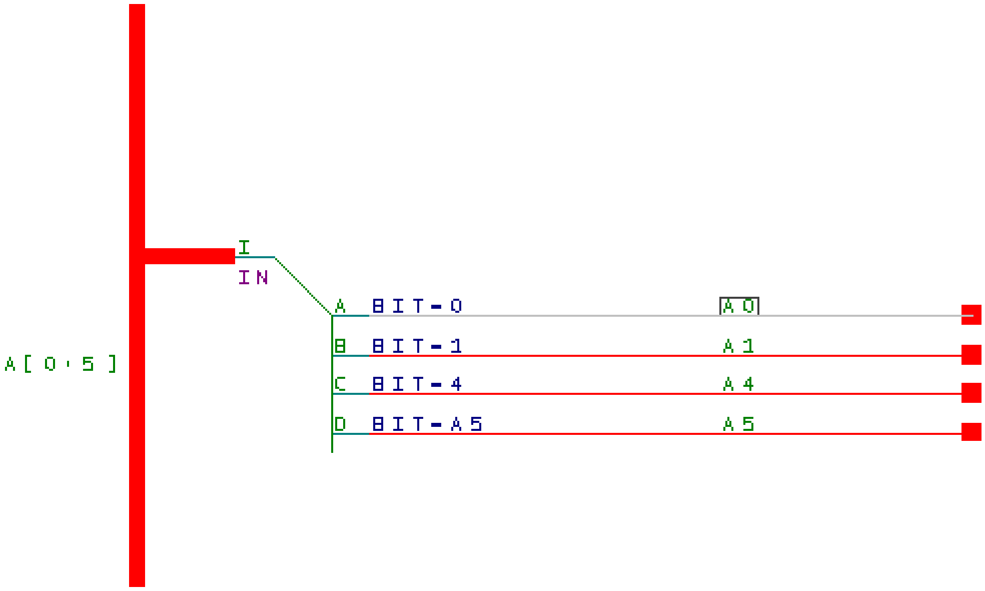


Figure 3: Same Notation on Different Buses

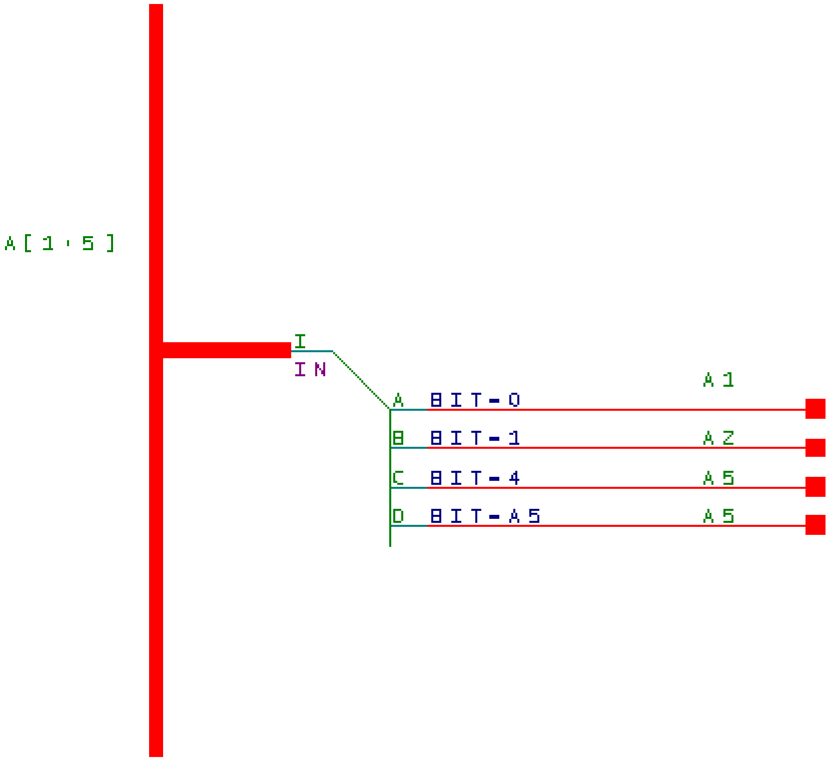


Figure 4: Same Notation on Different Buses Yields Different Signals

If you changed the bus index of the last signal of the bus ripper (in Figure 4), from `A5` to `5` (from net name to positional notation), ViewDraw generates the following message (when you save your design) because you have exceeded the range of the bus.

```
Checking Block tk1, Sheet2 ...
Wirelist created using version 5.2
Sch_Error -- Trying to rip bit 5 from label A[1:5].
1 error and 0 warnings on schematic tk1.2.
Check complete.
*********************************
```

See Also

Bus Ripper Symbols

Bus Rippers

Bus Rippers and Bundles

Connecting a Bus Ripper to a Bus

Creating a New Schematic

Net Aliasing

Resolving Net Names

Using Bus Rippers

Bus Rippers and Bundles

At a higher conceptual level of grouping signals (than a bus) is a bundle. ViewDraw supports ripping signals from bundles in the same fashion as in ripping signals from buses.

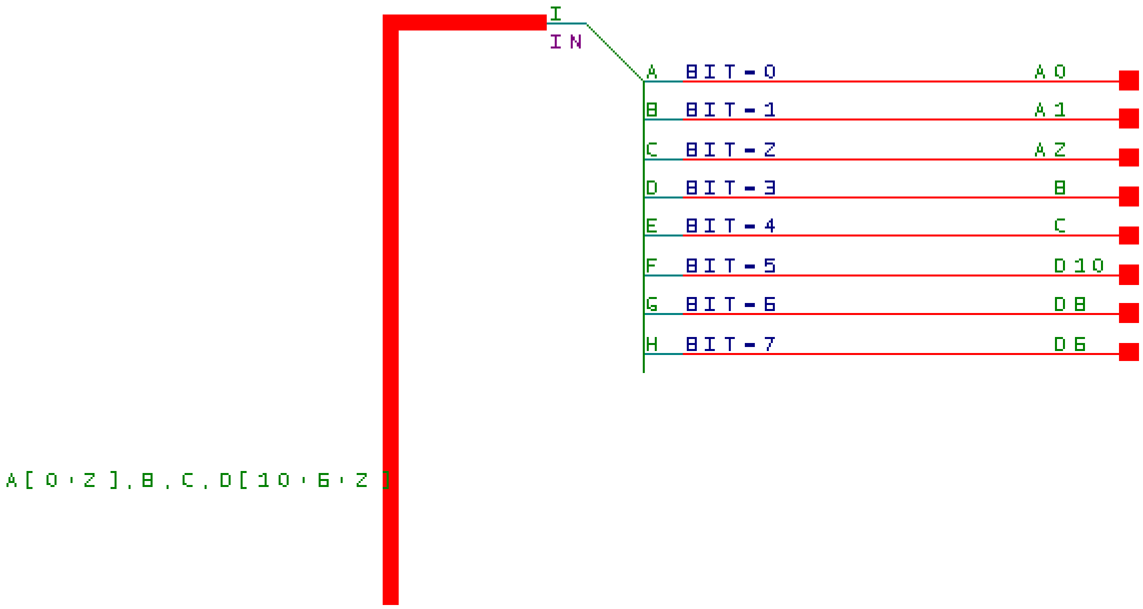


Figure 5: A Bus Ripper Pulling Signals from Different Sources

Consider the bundled signals shown in Figure 5. The 8–bit bus ripper pulls nets from two buses, A and D, along with two individual nets, B and C.

See Also

Bus Ripper Symbols

Bus Rippers

Connecting a Bus Ripper to a Bus

Creating a New Schematic

Extracting Signals From a Bus

Net Aliasing

Resolving Net Names

Using Bus Rippers

Net Aliasing

Explicit Net Aliasing

Using bus rippers, you can create an alias net name for a signal by labeling it. Without an intervening bus ripper, the net that you extract from a bus can only have one net name (see *Implicit Net Aliasing* for the exception). For example, Figure 6 shows two signals (A2 and A3) extracted from an 11–bit bus (A[0:10]). Because you must identify these nets for extraction, by the net name, you only have a single reference to each signal (the net names A2 and A3, respectively). There is no way to assign an alias net name.

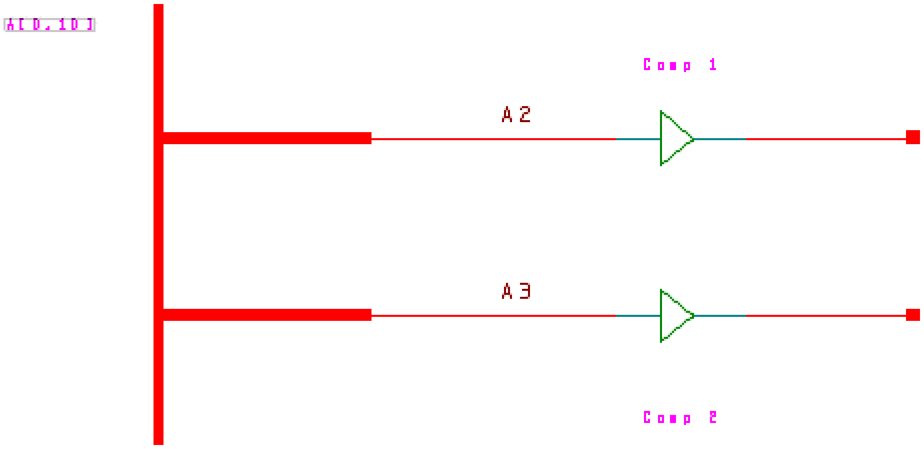


Figure 6: Extracting Signals From a Bus Without a Ripper

Figure 7 shows the same two signals extracted from the same bus with an intervening bus ripper. Note that positional identifiers are used instead of net names. Because the initial index of the bus starts at 0, positional identifiers 2 and 3 are equivalent to net names A2 and A3, respectively.

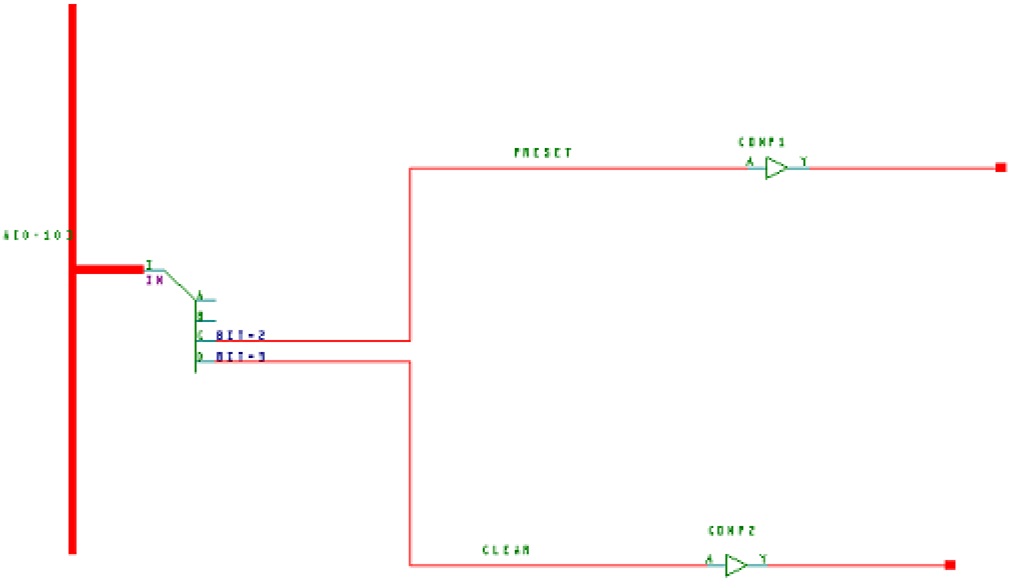


Figure 7: Extracting Signals From a Bus Using a Ripper

With an intervening bus ripper, you can create a net alias for each signal. Therefore, the two signals in Figure 7 each have two references (A2 and PRESET) for the net connected to Comp1, and (A3 and CLEAR) for the net connected to Comp2. Therefore, each of these nets can be referenced by either of two net names in subsequent operations.

When you choose **File** => **Save + Check**, ViewDraw generates the following message. Note the alias names for each net.

```
Checking Block a, Sheet 1 ...

Wirelist created using version 5.2

Sch_Info - Net PRESET has alias of A2

Sch_info - Net CLEAR has alias of A3

0 errors and 0 warnings on schematic a.1.

Check complete.

****************************
```

Implicit Net Aliasing

With bus ripper functionality enabled, RIPPERS_ON = 1, you can create an alias net name for a net which is connected to a global symbol pin (VDD, GND, etc.)

A net becomes global when it connects to a pin symbol that has the NETNAME attribute (NETNAME=GND, NETNAME=VDD, etc.)

For example, Figure 8 shows a bus going to a RAM chip with the highest–ordered bit pulled to ground. Because GND is a global signal, this net can be referenced by either RAM63 or GND.

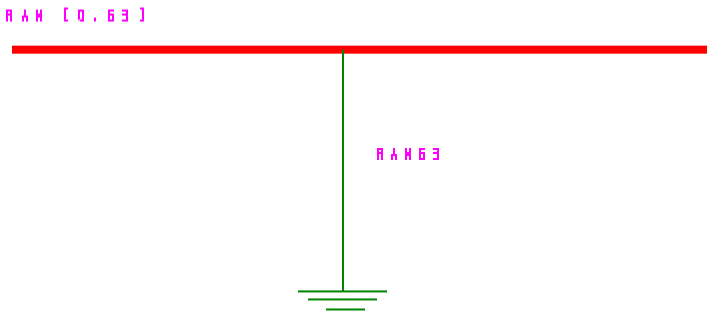


Figure 8: Net Aliasing Without a Bus Ripper

Bus ripper functionality must be enabled, RIPPERS_ON = 1, when aliasing net names that are connected to global pin symbols. ViewDraw generates a warning when you try to specify an alias net name for a global pin symbol when bus ripper functionality is disabled.

See Also

Bus Ripper Symbols

Bus Rippers

Bus Rippers and Bundles

Connecting a Bus Ripper to a Bus

Creating a New Schematic

Extracting Signals From a Bus

Resolving Net Names

Using Bus Rippers

Resolving Net Names

Although a net can have many names (with bus ripper net aliasing), only a single logical net exists in the design database. ViewBase (or ViewBase applications such as the **vsm** netlister) decides the primary net name for this logical net, based on a set of rules. ViewBase also retains alias net names associated with the primary net name through the NET_ALIAS attribute. Therefore, ViewBase maintains all names for a net and makes this information available to all tools that access the logical design database.

Figure 9 and Figure 10 show a design spread across two sheets. Each sheet contains a 5–bit bus: `Y` for sheet 1 and `Z` for sheet 2.

- The net connecting to Comp1 has two net names (`Y1` and `SET`). When you select this net, **SET** displays as the net name in the lower–left corner of the schematic window.
- The net connecting to Comp2 has one netname (`Y4`). When you select this net, **Y4** displays as the netname in the lower–left corner of the schematic window if Values or Name is on.

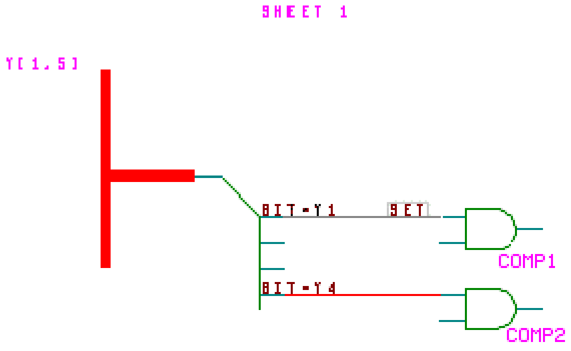


Figure 9: Sheet 1

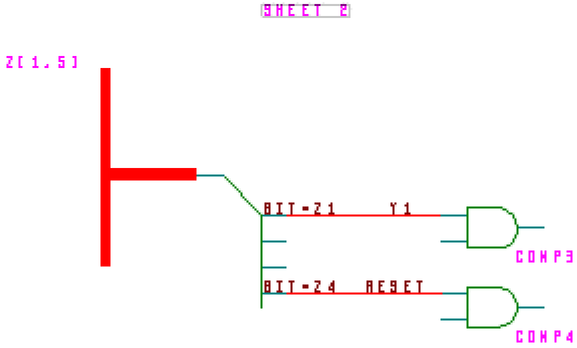


Figure 10: Sheet 2

Resolving Netnames Across Sheets

- The net connecting to Comp3 has three net names (`Z1`, `Y1`, and `SET`). When you select this net, **Y1** displays as the net name in the lower–left corner of the schematic window.
- The net connecting to Comp4 has two net names (`Z4` and `RESET`). When you select this net, **RESET** displays as the net name in the lower–left corner of the schematic window.

It is important to realize that the net connecting to Comp1 of Sheet 1 and Comp3 of Sheet 2 is logically the same net; although, with net aliasing, ViewDraw graphically shows these nets as different signals.

ViewBase, in turn, would collapse these aliased nets (including any associated attributes) into a single net. ViewBase would then determine the primary name of this net.

In ViewDraw, you can assign the CHOSEN=THIS attribute to a net to explicitly identify a primary netname to ViewBase.

See Also

Bus Ripper Symbols

Bus Rippers

Bus Rippers and Bundles

Connecting a Bus Ripper to a Bus

Creating a New Schematic

Extracting Signals From a Bus

Net Aliasing

Using Bus Rippers

Bus Ripper Symbols

This section covers the anatomy of a bus ripper symbol, default bus ripper symbols from the Builtin library, and how to customize a bus ripper symbol to meet your own needs.

A bus ripper is a ViewDraw symbol with special attributes. The bus ripper symbol consists of pins that will connect to signals that you wish to extract from the bus, and a single pin that you use as input to the bus ripper. The 4–bit bus ripper symbol shown in the following figure has five pins: one pin to connect to the bus (`I`); the other four pins to extract signals from the bus (`A,B,C,D`).

For bus rippers in ViewDraw, the default behavior is that the port A maps to the left most bit. For example if the bus is Addr [0:3] then A = Addr [0]. Conversely, if the bus is Data [3:0] then A = Data [3].

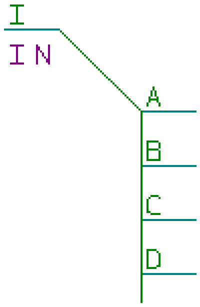


Figure 11: The Bus Ripper Symbol

The bus ripper symbol also has three attributes:

- The SYMTYPE attribute
  The SYMTYPE=RIPPER attribute (unattached) designates the symbol as a bus ripper.
- The WHICHRIPPED attribute
  The WHICHRIPPED attribute (unattached) specifies a symbol pin that connects the bus ripper symbol to the bus.
- The BIT attribute
  The BIT attribute (attached) on a symbol pin specifies which signal to extract from the bus.

Library Symbols

The Builtin library contains 4–, and 8–bit bus ripper symbols (4b_rip, and 8b_rip, respectively). The next section explains how to customize library bus rippers and how to create a new bus ripper symbol.

Customizing Bus Ripper Symbols

You can customize the bus ripper symbols from the Builtin library. You can also create your own bus ripper symbol. Creating a New Symbol explains how to create and modify symbols. The following is required of bus ripper symbols:

- Specify a SYMTYPE=RIPPER attribute on the symbol.
- Label all symbol pins.

Bus rippers from the Builtin library have pins labeled `A1` through `A`*n*. You can choose your own pin labeling scheme, as long as you label all the symbol pins.

- Specify a WHICHRIPPED attribute on the symbol (unattached) with an argument of the pin name that you designate to connect the bus ripper symbol to the bus.
- Specify a BIT attribute on each pin designated to extract signals from the bus.

If you are working with a bus larger than 8–bits, rather than creating a symbol from scratch, it is easier to copy the Builten 8–bit bus ripper symbol, add the necessary pins, then save the new symbol. You do not have to use all pins of a bus ripper symbol. Use only those pins that are necessary to accommodate the signals being extracted from the bus.

See Also

Bus Rippers

Bus Rippers and Bundles

Connecting a Bus Ripper to a Bus

Creating a New Schematic

Extracting Signals From a Bus

Net Aliasing

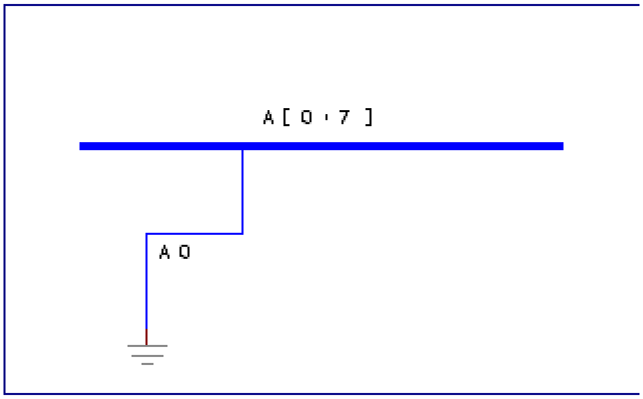
Resolving Net Names

Using Bus Rippers

Grounding Individual Nets of a Bus

ViewDraw uses bus ripper components to alias individual nets of a bus to ground.. If you wish to accomplish the same task without enabling bus rippers, you can pull individual nets from the bus and connect them to the GND symbol. This approach can be used for either PCB or simulation.

Example:
In the example below, net A0 and ground are aliased together. Therefore, net A0 is treated as part of the ground net in the ViewDraw database.



See Also

Bus Ripper Symbols

Bus Rippers

Bus Rippers and Bundles

Connecting a Bus Ripper to a Bus

Creating a New Schematic

Extracting Signals From a Bus

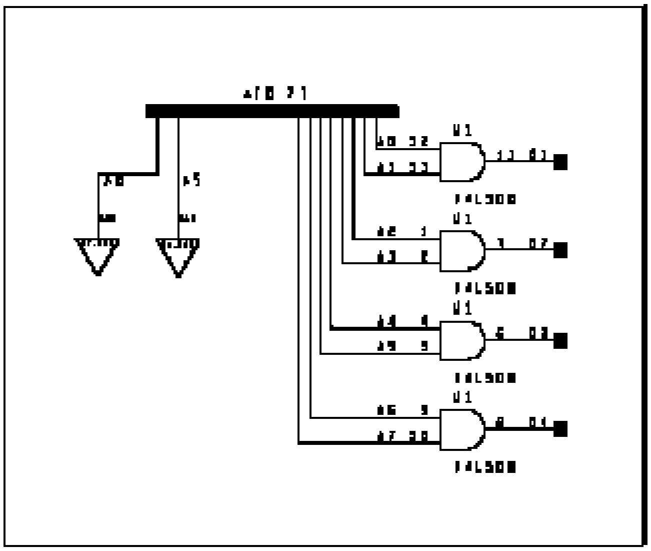
Net Aliasing

Resolving Net Names

Using Bus Rippers

Grounding Nets of a Bus

As shown in the following figure, you can place this hierarchical MY_GND symbol on your schematic and connect it to the net A0.



The net attached to the MY_GND pin can have the label A0 because the global net label GND exists in the lower level of hierarchy.

Label the bus A[0:7]. If you have a case where you want additional bus segments connected to GND, then you must break each additional net segment off the main bus, and each net must be attached to a MY_GND component and labeled with the individual bit label. For example, the nets may be labeled A0 and A5 if bits 0 and 5 are to be connected to ground.

Alternate Approach

An alternate approach to directly connecting a bus net segment to ground in the schematic is to force the signals low, for simulation purposes only. To accomplish this, during a simulation, simply force the signal low. For the previous example, a simulation command file may include the following line:

```
L A0 A5
```

This would be done instead of the recommended method described previously.

This method will not connect the net segments to ground in any other applications or interfaces provided by Innoveda, for example PCB interfaces.

Connectivity

This section discusses connectivity features in ViewDraw — specifically Pass-Through Pins and Routing.

Click on the topic you want more information about:

Pass-Through Pins

Routing Modes

Pass-Through Pins

Pass-through pins are two or more pins on a symbol with the same label. You typically place these pins on opposite sides of a symbol. A net connected to one of the pins routes straight through the component to the other pin(s) as shown below.

Designs using pass-through pins must be flattened before they are sent to a down-stream simulator. Do not use pass-through pins for hierarchical netlisting, such as Export1076, or if the pins exist at the MODULE level, such as Zycad, C models, Verilog, LMC models, VHDL, or compiled code.

You can use pass-through pins to build neat arrays of components without having to route nets around (avoidance routing) or over (straight routing) the component. Specific applications include memory, PLDs, connectors, and analog circuit design.

Labels and Attributes

Label a net connected to a pass-through pin only once. A pass-through pin cannot connect two nets that have different labels.

ViewDraw considers an attribute on any pass-through pin to be on all pins of that name. Place all attributes for a specific pass-through pin on only one of the pins.

Wirelist Interpretation

Pass–through pins only represent a graphical shorthand. Logically (in the logical database), only one pin exists. The extra (graphical) pins only appear in the symbol and schematic files.

ViewDraw writes both pins to the symbol and the schematic files. The **Tools - Check** command removes redundant pins for the logical database and renames nets associated with pass-through pins. ViewDraw uses the lowest numbered net internal name or label for the entire set of nets. The pin order only consists of the logical pins. ViewBase and other logical applications do not recognize the additional pins.

To get the **Tools - Check** command, you must add the check utility to the Tools menu using the Customize menu option. Refer to the Customizing the Tools Menu section for more information.

See Also

Connectivity

Routing Modes

All connectivity is formed in ViewDraw with the specification of a routing mode. The routing modes are:

- Avoidance
- Orthogonal
- Straight

Changing the Routing Mode

The **viewdraw.ini** file specifies the routing mode with the ROUTE keyword. You can change the routing mode during ViewDraw operation with the Route Mode setting of the Nets tabs on the Project Settings dialog box. (Use the Project Settings command to get to the Project Setting dialog box.)

Avoidance Routing

Avoidance routing specifies an automatic connection between two points that avoids components. Specifying intermediary points along the connection creates a more distinct specification of the path of the connection. The distance between components and the connection is the avoidance distance.

Orthogonal Routing

Orthogonal routing specifies a 90 degrees connection between two specified points. Orthogonal routing begins with a horizontal or vertical orientation depending on the point of origin as follows:

- From a joint — begins in a horizontal or vertical direction following the cursor movement (left or right, horizontal — up or down, vertical)
- From a pin — automatically begins in a horizontal or vertical direction, depending on the orientation of the pin
- From a point on a net or bus segment — begins in a direction perpendicular to the orientation of the segment

Orthogonal routing does not allow the specification of a connection that passes through a component.

Straight Routing

Straightrouting specifies a straight connection between two points. This "as is" form of routing can overlap components and existing connections. Any incidental crossing of connectivity from routing does not imply a connection.

Snap–to–pin Mode

You can also specify routing to snap to the closest pin with the SnapToPin setting on the Nets tab of the Project Settings dialog box. Use the Project Settings command to get to the Project Settings dialog box. This places the SNAPTOPIN keyword in the **viewdraw.ini** file. When the connection is formed, the route snaps to the nearest pin from either a point specified inside (on) or outside (off) the component.

If SNAPTOPIN is on, the connection snaps to the closest pin if the cursor is placed inside the component. If SNAPTOPIN is off, the connection snaps to a pin if the cursor is placed within the selection distance of a pin.

See Also

Connectivity

Avoidance Distance

ViewDraw automatically staggers a connection by the avoidance distance as the route hugs to the components or other connections. You specify the avoidance distance with the Avoidance setting of the Project - Settings command or with the ADISTANCE keyword in the **viewdraw.ini** file.

Vector Expansion

Vector Expansion

Component Arrays This section discusses expansion techniques in ViewDraw — specifically Component Arrays and Vectorized Labels.

Click on the topic you want more information about.

Component Arrays

Vectorized Labels

Miscellaneous Advanced Features

Unique Labels

The **viewdraw.ini** file specifies unique labels either on or off with the UNIQUE_LABEL keyword. This keyword turns on or off the creation of unique labels from internal names when components, nets, buses, or pins are copied, arrayed, or pasted from the buffer.

Unique Label Format

When unique labels are on, ViewDraw creates the new labels as follows:

- Components — $*sheet*I*uid* (for example, $1I350)
- Pins — $*sheet*P*uid* (for example, $1P247)
- Nets — $*sheet*N*uid* (for example, $1N335)
- Buses — $*sheet*N*uid* (for example, $1N325)

Where *sheet* is the sheet number of the active window and *uid* is the unique number assigned by the system. Refer to the section Internal Names.

Note: For PCB layout, the $ may violate the labeling conventions for some PCB layout systems. When you run PCBFWD, it resolves this conflict automatically for you. Refer to the PCB for more information about resolving labeling conflicts.

See Also

Bus Ripper Symbols

Bus Rippers

Bus Rippers and Bundles

Connecting a Bus Ripper to a Bus

Miscellaneous Advanced Features

Net Aliasing

Resolving Net Names

Using Bus Rippers

Open Hyperlink - Associating Objects With URL's

You can associate ViewDraw schematics, components, and symbols with Web pages. For example, you can associate a component with a data sheet, or a design specification with a schematic. You do this by using a special attribute that enables a new popup menu item.

To implement Open Hyperlink functionality:

- Place the Hyperlink attribute unattached on a schematic, symbol, or component.

Rules:

- The format of the attribute must be NAME=[NICKNAME;]URL.
- The NICKNAME in the value field is optional. If you use it, you must separate it from the URL with a semicolon (;).

- As with any ViewDraw attribute, the Name and Value fields must be upper case. If you enter them in lower case, ViewDraw will force them to upper case. In most situations, this will be acceptable. Therefore, if at all possible, you should use upper case URL's.

Important: Using lower or mixed case attribute values can cause problems with third party tools, and we strongly recommend that you avoid doing so. However, if you must use a lower or mixed case URL, we provide two methods for doing so. Click on a topic below for more information.

Encode the values to be lower case. Under most circumstances, you should use this method, because it affects only the attribute value that you encode.

Enable lower case values in ViewDraw.ini for the hyperlink attribute(s) you create. This is known as Case Preservation.

Warning: You should use Case Preservation with caution. If you enable Case Preservation and write out your design, ViewDraw will increment the revision number of the design database. This database version will not be compatible with earlier versions of ViewDraw. To decrement the revision level, you must disable Case Preservation and write the design out again. Once you enable Case Preservation, you will have to do this if you want to use the database with earlier versions of ViewDraw, or with any downstream, third party tools which do not support it. This fact is often sufficient reason to choose not to use Case Preservation.

Examples:

```
VENDOR=HTTP:// US;WWW.COMPANY.COM
```

```
DATA=FTP://FTP.COMPANY.COM
```

```
MAIL=MAILTO:PERSON@COMPANY.COM
```

3. Right-click to access the Open Hyperlink pulldown. Depending on the object, you will right-click a different part of the screen.

- On the schematic, right-click any empty area of the schematic.
- On the component, right-click the component.
- On the symbol, right-click any empty area outside the bounding box of the symbol.

4. From the pulldown, select Open Hyperlink. You will see a list of the hyperlinks that you have created. If you have used a nickname, the nickname will be listed. If not, the attribute name will be used.
5. Select the hyperlink you want from the pulldown.
   ViewDraw will open the URL you have specified.

Enabling Mixed-case Attribute Values - Case Preservation

Introduction to Case Preservation

Older versions of ViewDraw always required attribute names and values to be upper case. If you did not enter them in upper case, ViewDraw forced them to upper case and stored them in upper case in the database and wirelist.

From this version of ViewDraw, you can select how ViewDraw treats the case of attribute values. You do this using a new functionality called Case Preservation. When you enable Case Preservation, ViewDraw preserves the case of selected attribute values as you enter them, rather than forcing them to upper case.

Note: Attributes Names and Labels will still be forced to upper case, and may not be changed.

You control Case Preservation by using three new keywords that are found in ViewDraw.ini. You use these keywords to define the default behavior, as well as to allow or disallow mixed case for individual attributes.

The new keywords are:

```
PRESERVE_CASE
```

```
ALLOW_VALUE_MIXED
ALLOW_VALUE_MIXED_FILE
```

You change the value of these keywords by editing ViewDraw.ini with any text editor, or by using automation methods.

Important Restrictions:

Keep the following restrictions in mind when you decide if you want to enable Case Preservation.

You should use Case Preservation with caution. If you enable Case Preservation and write out your design, ViewDraw will increment the revision number of the design database. This database version will not be compatible with earlier versions of ViewDraw. To decrement the revision level, you must disable Case Preservation and write the design out again. Once you enable Case Preservation, you will have to do this if you want to use the database with earlier versions of ViewDraw, or with any downstream, third party tools which do not support it. This fact is often sufficient reason to choose not to use Case Preservation.

Only attribute values can be controlled. Attribute names and labels must still be upper case.

You should not specify mixed case for values that refer to label names.

The following PCB tools support Case Preservation:

Allegro: Supported for all attributes except pin number (#) and REFDES for version 13.5 or greater.

Mentor Graphics: Fully supported for version C.2 or greater.

Pads: Supported for all attributes except pin number (#) and REFDES for version 3.0 or greater.

Visula: Fully supported for version 5.0.2 or greater.

Other third party tools may not support Case Preservation. Check with your tool vendor.

The Case Preservation Keywords

The keywords used in Case Preservation are PRESERVE_CASE, ALLOW_VALUE_MIXED, AND ALLOW_VALUE_MIXED_FILE.

```
PRESERVE_CASE
Syntax: PRESERVE_CASE <0/1>
```

Defaults to 0.

If set to 0, Case Preservation is disabled. All other keywords are ignored, and all values are forced to upper case. This is identical to the behavior of previous versions of ViewDraw

If set to 1, Case Preservation is enabled. All other keywords are activated.

ALLOW_VALUE_MIXED

Syntax: ALLOW _VALUE_MIXED <attribute name>

Ignored unless PRESERVE_CASE is 1.

Specifies which attribute(s) are allowed to have mixed-case values.

Requires one instance of the keyword for each attribute.

ALTERNATE: ALLOW_VALUE_MIXED_FILE

```
Syntax: ALLOW_VALUE_MIXED_FILE <filename>
```

<filename> is the name of a user-created text file that contains a list of attributes whose values are allowed to be mixed-case.

You can give a complete path as the <filename> argument. If you do not, ViewDraw assumes that the file is located in the Project directory.

Using Case Preservation

The following is a general procedure for using Case Preservation. You adapt this general procedure to your specific requirements. For specific ways that you can adapt Case Preservation to your requirements, see the Case Preservation Examples section.

Note: Because Case Preservation has so many ramifications, you must work in the ViewDraw.ini file, or using Automation methods, rather than using the Project >Settings Dialog Box. With this more centralized method, it becomes more difficult for one user to change the Case Preservation status for the design. This helps you to maintain tighter control over use of this feature.

Perform the following steps, in any order, to apply Case Preservation to your design:

1. Enable Case Preservation.
2. List allowed mixed-case attributes.

Step 1: Enabling Case Preservation:

Enable case preservation by setting the keyword PRESERVE_CASE to 1. Case Preservation must be enabled for any of the other Keywords to be active. You set the keyword by one of the following methods:

Viewdraw.ini:

1. Edit ViewDraw.ini.
2. Change the value of THE PRESERVE_CASE keyword to 1.

ViewDraw Automation:

1. Use the ParamSetMode method. This is a method of the application.
2. Use an index of VMD_PRESERVE_CASE and a NewValue of TRUE.

Example: Application.ParamSetMode(VDMD_PRESERVE_CASE)=TRUE

Step 2: Listing allowable mixed-case or lower-case attributes

Use the keyword ALLOW_VALUE_MIXED [_FILE] to list attributes whose values you will allow to have their case preserved. By default, this list is empty. You list allowed mixed-case attributes by one of the following methods:

ViewDraw.ini

1. Edit ViewDraw.ini.
2. For each attribute you want to add to the list, add a line to ViewDraw.ini that reads:

```
ALLOW_VALUE_MIXED <attribute name>
```

Alternative:

1. Create a text file that contains a list of the attributes whose values you will allow to have their case preserved. This file must contain a list of attribute names, one name per line.
2. Edit ViewDraw.ini

3. Add a line to ViewDraw.ini that reads:

```
ALLOW_VALUE_MIXED_FILE <filename>
```

where <filename> is the name of the text file you created in step 1. You can use a complete path to the filename. If you do not, ViewDraw will assume that the file is in your project directory.

ViewDraw Automation

1. Use the AddListName method. This is a method of the application.
2. For the parameter Which, use a value of MIXED_LIST
3. For the parameter New, give the name of the attribute.

Example: Application.Param.AddListName(MIXED_LIST, "MYATTR")

File Format Changes with Case Preservation

When you turn on Case Preservation, this will be reflected in both the Wirelist and Schematic/Symbol files.

Changes in the Wirelist (WIR) file:

Three lines in the WIR file will be affected by turning on Case Preservation:

The Header - Version will be incremented to at least 5.3.

The V line - This will match the version number in the header.

The Q line - This will reflect the addition of "Case" to ViewDraw options.

The code snippet below shows a typical WIR file for a design named testt that has Case Preservation enabled.

```
|Wirelist created using version 5.4.

V 5.4

K 427286292700 testt

DW testt

Q Case
```

Changes in the Schematic/Symbol files:

Two lines in the schematic or symbol file will be affected by turning on Case Preservation:

The V line - The version will be incremented to at least 5.3.

The F line - This will reflect the addition of "Case" to ViewDraw options.

The code snippet below shows a typical Schematic/Symbol file for a design named testt that has Case Preservation enabled.

```
V 53

K 427286292700 testt

F Case
```

Case Preservation and Previous Database Revisions

When ViewDraw reads in a database, it checks for the revision. If that revision is 5.3 or less, ViewDraw considers it to be an "old" database. You can configure how mixed case or lower case attribute values will be handled when ViewDraw reads in an "old" database. You do this with two environment variables, VL_UPPER_OLD_DB and VL_CASE_WARNING. There are three options for handling "old" database revisions.

1. Configure ViewDraw to force all attribute values to upper case and generate a warning.
2. Configure ViewDraw to force all attribute values to upper case and suppress a warning.
3. Configure ViewDraw to preserve the case of attribute values.

Forcing to upper case and generating a warning:

To force all attribute values to upper case and generate a warning, add the VL_UPPER_OLD_DB and VL_CASE_WARNING variables to your environment, and set them true.

Forcing to upper case and suppressing a warning:

To force all attribute values to upper case and suppress a warning, add only the VL_UPPER_OLD_DB variable to your environment and set it true.

Preserving the case of attribute values:

To preserve the case of attribute values as they are read in, you must ensure that VL_CASE_WARNING and VL_UPPER_OLD_DB are not present in your environment. If you have previously added and set these variables you must remove them.

Case Preservation Examples

You can adapt the general Case Preservation procedure to your specific needs. The examples below cover some of the more common scenarios

Example 1: Allowing a mixed-case value for only a single attribute, XYZ:

1. Edit ViewDraw.ini
2. Set PRESERVE_CASE to 1.
3. Add the following line to Viewdraw.ini:

```
ALLOW_VALUE_MIXED XYZ
```

Only XYZ will be allowed to have mixed case. If you attempt to enter a mixed-case value for any other attribute, it will be forced to upper case.

Example 2: Allowing mixed-case values for only a certain group of attributes, ABC, DEF, GHI:

1. Edit ViewDraw.ini
2. Set PRESERVE_CASE to 1
3. Create a file named, for example, mixed.dat. The contents of the file will be:
   ABC
   DEF
   GHI

```
Add the following line to ViewDraw.ini
ALLOW_VALUE_MIXED _FILE mixed.dat
```

Alternative:

1. Edit ViewDraw.ini
2. Set Preserve Case to 1

```
Add the following lines to Viewdraw.ini
ALLOW_VALUE_MIXED ABC
ALLOW_VALUE_MIXED DEF
ALLOW_VALUE_MIXED GHI
```

Only attributes ABC, DEF and GHI will be allowed to have mixed-case values. If you attempt to enter a mixed-case value for any other attribute, it will be forced to upper case.

Introduction to OATs

OATS -- What Are They?

Occurrence Attributes (OATs) are a special type of ViewDraw attribute that allow you to develop true hierarchical designs. You assign OATs to components, component pins, and nets.

You use occurrence attributes when there are multiple instances of a composite component throughout the design hierarchy. Each instantiation of that component has the same underlying schematic which defines its logic, thereby, *effectively* multiplying the number of times that you use the underlying schematic within the design. An occurrence attribute is defined using the exact path to that occurrence.

An Important Note About OATS:

To use OATs, you should open the project at the top level and push through the levels of hierarchy. OATs are associated from the top–level schematic to the hierarchical level of the occurrence.

OATs -- What Can They Be Added To?

With OATs, you can instantiate many designs whose underlying designs are identical except for instance-specific attributes like REFDES and Pin Numbers (#). OATs can be added to components, component pins, and nets.

A component OAT, if it exists, overrides a symbol or component level REFDES attribute and is used to add a new attribute to a single occurrence of a component.

A component pin OAT , if it exists, overrides a symbol or component level PinNumber (#) attribute or it could add a new attribute to a single occurrence of a component pin.

A net OAT , if it exists, overrides a net level attribute, such as DELAY or CAPACITANCE, or it could add a new attribute to a single occurrence of a net.

OATs -- Why Use Them?

You use OATs for creating hierarchical designs that require a unique mapping to a flattened representation. OATs provide for back annotation (update) from flattened PCB databases to a hierarchical schematic database.

Using the unique hierarchical mappings that OATs offers, the same underlying schematic can service many instances of the same object throughout the design. Since you can make occurrence specific changes to an object a schematic underlying your top-level design via OATs, you can reuse this same schematic without having to duplicate the non-attributes objects on the schematic to form addition hierarchical blocks in the design.

A Typical Use for OATs:

You assign a reference designator (REFDES) to component instances. When you prepare for PCB layout, you package the design into physical packages and each package can include multiple components or parts. You assign a unique REFDES to each package. This REFDES is then annotated on each component or part in the design hierarchy.

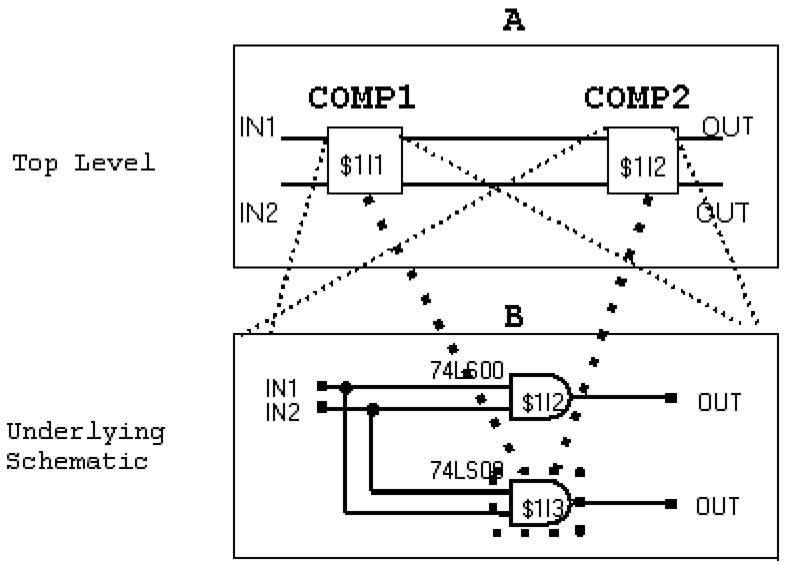
Suppose you have a composite component instantiated twice. Underneath it are 2 instances of an and gate. And finally, suppose only one part is permitted in the package. In this case, the REFDES attributes on the ands of the underlying schematic's components cannot be attached as a regular attribute. This would result in a duplication of the the REFDES on the schematic of the each of these hierarchical components. When you assign the REFDES as an OAT, the ands for the hierarchical component can have unique REFDES values dependent upon each hierarchical path.

OAT Placement

Attribute placement requires a placeholder in the schematic. This information is not stored in the .oat file with the OAT attribute. Therefore, the placement cannot be anything other than the default location unless there is a symbol, component, or net level attribute of the same name to act as a placeholder for the OAT. This can result in problems for a unique OAT attribute. If the default placement is unacceptable when printing your schematic, either make the attribute invisible or add a lower-level attribute of the same name with no value to act as a placeholder.

OATs -- An Example

The following design shows two top-level components (COMP1 and COMP2) that use the same underlying schematic.



Schematic B serves as the underlying schematic that defines the logic of COMP B. COMP B is instantiated twice on the top level schematic (COMP1 and COMP2). The logic at the lower level is effectively used twice at the higher level.

In effect, object COMP1 and COMP2 at the top level are defined by the same underlying schematic. If you push down from the top of the design hierarchy through COMP1 and then again through COMP2, you get a different paths to the 74lS00.

What Happens when you add an Instance Attribute to the 74SL00 component?

When you add an instance attribute to the 74SL00 component on the underlying schematic, it affects all components in the design hierarchy that use that underlying schematic. Because COMP1 and COMP2 are defined by the same underlying schematic, an instance attribute on the 74SL00 component on Schematic B affects both COMP1 and COMP2.

What Happens when you add an OAT to the 74SL00 component?

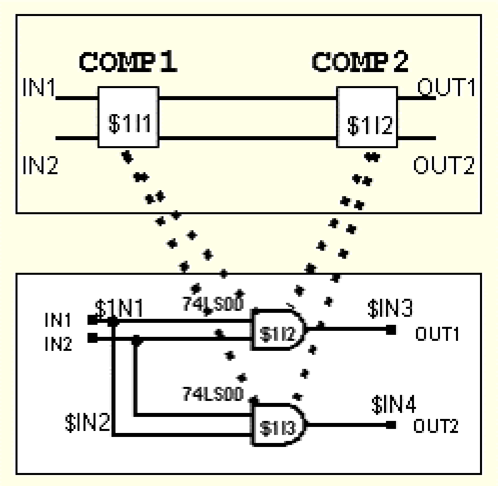
When you add an occurrence attribute to 74SL00 component at the bottom of schematic B, that attribute affects only the component in the hierarchy that you pushed down from. If your path to the 74SL00 was through COMP1, the OAT affects only COMP1. ViewDraw forms a path using a unique hierarchical description to associate with the OAT you added. In this example, the unique hierarchical path is $1I1\$1I3.

See Also

Forming a Hierarchical Path

Forming a Hierarchical Path

ViewDraw uses unique hierarchical descriptions to associate attributes with a particular occurrence. The occurrence is comprised of the unique identifier (UID) of each component in the path to the occurrence, followed by the identifier of the selected component, component pin, or net at a lower level of a hierarchical design. Therefore, each component, component pin, or net has a unique hierarchical path which describes where that object occurs in the design relative to the top–level schematic.



Paths to an Occurrence

Pushing down from COMP1 or COMP2 at the top of the hierarchy into the selected component's underlying schematic yields the following hierarchical descriptions. Each description consists of the path of all levels to the occurrence plus the object's internal name.

- COMP1 to 1st 74LS00 1I1\$1I2
- COMP1 to 2nd 74LS00 $1I1\$1I3
- COMP2 to 1st 74LS00 $1I2\$1I2
- COMP2 to 2nd 74LS00 $1I2\$1I3

Nets and pins use labels (if defined) instead of internal names.

- COMP1 to net IN1 $1I1\$IN1
- COMP1 to net IN2 $1I1\$IN2
- COMP1 to net OUT1 $1I1\$IN3
- COMP1 to net OUT2 $1I1\$IN4

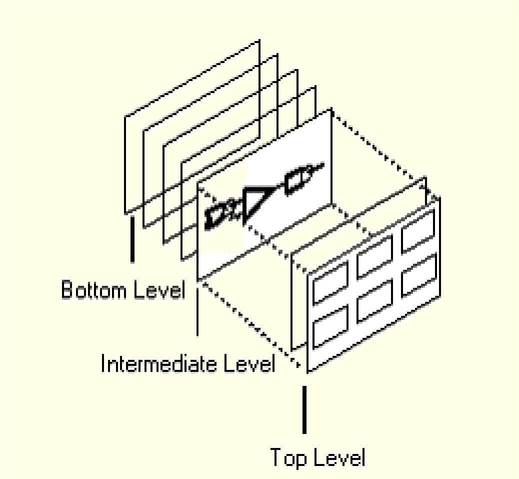
- COMP2 to net IN1 $1I2\$IN1
- COMP2 to net IN2 $1I2\$IN2
- COMP2 to net OUT1 $1I2\$IN3
- COMP2 to net OUT2 $1I2\$IN4
- COMP1 to input pin 1 on
  1st 74LS00 (not shown) $1I1\$1I2.A
- COMP1 to input pin 2 on
  1st 74LS00 (not shown) $1I1\$1I2.B
- COMP1 to output pin on
  1st 74LS00 (not shown) $1I1\$1I2.Y
- COMP1 to input pin 1 on
  2nd 74LS00 (not shown) $1I1\$1I3.A
- COMP1 to input pin 2 on
  2nd 74LS00 (not shown) $1I1\$1I3.B
- COMP1 to output pin on
  2nd 74LS00 (not shown) $1I1\$1I3.Y
- COMP2 to input pin 1 on
  1st 74LS00 (not shown) $1I2\$1I2.A
- COMP2 to input pin 2 on
  1st 74LS00 (not shown) $1I2\$1I2.B
- COMP2 to output pin on
  1st 74LS00 (not shown) $1I2\$1I2.Y
- COMP2 to input pin 1 on
  2nd 74LS00 (not shown) $1I2\$1I3.A
- COMP2 to input pin 2 on
  2nd 74LS00 (not shown) $1I2\$1I3.B
- COMP2 to output pin on
  2nd 74LS00 (not shown) $1I2\$1I3.Y

OATs Visibility and Placeholder Attributes

An OAT that is added will assume the visibility characteristics of the corresponding attribute with the same name (that has a default value). For example, on a component which has an attribute named REFDES with a value of U? on the symbol, U1 on the component and U33 as an OAT value, the display characteristics like orientation will be inherited from the component attribute. This information is saved with the design. If an OAT exists without a component value, then the OAT shows up in an arbitrary location on the object until a default value is specified. If you do not want to see these placeholder values, go to the Project Settings, Attributes tab and uncheck the Display Placeholders option.

Understanding OATs Files and Hierarchy

The following design consists of seven levels of hierarchy. The following scenarios demonstrate using OATs at different levels and how ViewDraw saves them.



OATs and Hierarchy

Project Level OATs

ViewDraw stores OATs in the project database. Each project that uses OATs has a **.oat** file associated with it. Create (and update) the **.oat** file when you manually add an OAT (using the oattribute command-line command, double-clicking the owner object, or double-clicking on the attribute with the default value) or when you execute the **Tools -- Create REFDES** or **Tools -- PCB Netlist** commands. These commands save all occurrence attributes.

The **File** -- **Open** command reads the **.oat** file from disk and replaces OATs in memory. Executing **File** -- **Open** loads only one sheet of the schematic.

Note: To use OATs, you should open the project at the top level and push through the levels of hierarchy. OATs are associated from the top–level schematic to the hierarchical level of the occurrence.

For example, if you enter the design at the top–level schematic, push down to the bottom level schematic, select a component and assign an OAT, then the following is true:

- The OAT is known throughout the entire design
- The OAT is stored in the project .**oat** file
- You can access this OAT only by re–entering the project from the top schematic level and pushing down to the occurrence

You should always assign an OAT relative to the top–level schematic (the project level).

Intermediate Level OATs

You should always enter your project from the top-level. If you enter a project from an intermediate level, the OATs you create are associated with the project from that level down. The system stores any OATs that originate from an intermediate level in a separate .**oat** file.

Warning!:

When you enter a project from an intermediate level in the design hierarchy, OATs that you create are stored in an OAT file with that intermediate name. Since these are not automatically associated with some higher-level schematic (top-level), you **will not** see them when you enter the design from the top-level later on. Use the Oat Tools Propagate feature to bring these intermediate level OATs to the top level.

If you enter the design at an intermediate level schematic, push down to the bottom level schematic, and then select a component and assign an OAT, the following is true:

- The OAT is not known above the intermediate level
- The OAT is stored in a separate .oat file
- You can only access this OAT by re-entering the project from the intermediate level

How Do You Use Oats

Using OATs

You assign OATs by pushing into your design and selecting an occurrence of a component, component pin, or net. ViewDraw builds a hierarchical description that starts with the hierarchical level where you enter the design followed by all paths to the occurrence, followed by the occurrence itself — the selected component, component pin, or net. You then assign the OAT value. To do this, you double-click the object and select the Attribute Tab. Then, enter the OAT value in the OATs field and click OK.

An OAT that is added assumes the visibility characteristics of the corresponding attribute with the same name (that has a default value). If an OAT exists without a component value, then the OAT shows up in an arbitrary location on the object until a default value is specified

An Important Note About OATS:

To use OATs, you should open the project at the top level and push through the levels of hierarchy. OATs are associated from the top–level schematic to the hierarchical level of the occurrence.

There are 3 ways to add OATs:

- Using the oattribute command-line command
- Double-clicking the object and selecting the Attribute tab
- Double-clicking the attribute itself

Accessing OATs Features

All OAT commands apply to the selected component, component pin, or net. When you want to add or modify an OAT, use one of the following methods:

Once the component, component pin, or net is selected, use the right click to display the popup menu. Select **Properties** from the popup menu to do the following:

- Add an OAT
- Change an OAT name
- Change an OAT value
- Change an OAT string (name and value)

You can also use the following commands from the ViewDraw command line:

- To Add an Oat, use the attribute command.
- To get details about the design (including OAT details), use the oatsdetail command.

You can use the properties dialog box to add or change an individual OAT. If you want to add a common OAT to one or more objects, use the oattribute command. (The multiple objects must all be the same type; component, component pin, or net.)

Enabling OATs

The default mode for OATs is off. You do not have to enable OAT mode each time you wish to add an OAT. Once enabled, OAT mode remains active until you turn it off.

To Enable OATs in ViewDraw:

1. Close all open documents in ViewDraw, and select the **Project - Settings** command (or click).
2. In the Project tab, select the **OATs** option and click **OK**.
3. This turns on the OATs functionality.
4. Open your schematic.

To Disable OATs in ViewDraw:

1. Close all open documents in ViewDraw, and select the **Project - Settings** command (or click).
2. In the Project tab, deselect the **OATs** option and click **OK**.
   This turns off the OATs functionality.
3. Open your schematic.

Adding an OAT

Before you can add an OAT, you must have OATs enabled in your project. If you have not enabled OATs, refer to Enabling OATs for more information.

Remember that when you use OATs, you should open the project at the top level and push through the levels of hierarchy. OATs are associated from the top–level schematic to the hierarchical level of the occurrence. Once you have enabled OATs, follow these steps to add an OAT.

To Navigate through the design

1. Select hierarchical component on the higher–level schematic.
   You should always assign an OAT to an object on a lower–level schematic by navigating through the hierarchy from the top–level schematic (the project level).
2. Click on the component to select it.
3. With the object selected, right-click to display the shortcut menu.
4. Select the **Schematic** option from the pop up menu. (You can also use the .)
   You must select a component at each hierarchical level and use the **Schematic** option from the pop up menu repeatedly until reaching the desired sheet. ViewDraw builds a hierarchical path of all levels traversed to the sheet.
5. Choose the **psheet** *n* keyboard command to move to a selected sheet at the same level in the design. For example, **psheet 3** from the command line moves to sheet 3 at the same level in the design.

To Add the OAT and default value:

1. Click the owner object (net, component, or component pin) to select it.
2. Right-click and select **Properties** from the pop up menu. The **Properties** dialog box appears.
3. Select the **Attributes** Tab.
4. Set the appropriate visibility setting.

5. Enter the attribute name, value, and OAT
   For example: Name field: **REFDES**
   Value field: **U1**
   OAT field: **U22**
6. Click the **Set** button.
7. Click **OK**.

Modifying an OAT

To modify an OAT:

1. Navigate through the design to the component, component pin, or net that has the OAT you want to modify.
2. Double-click the component, component-pin, or net. The **Properties** dialog box appears.
3. Select the **Attributes** Tab.
4. Scroll through the list of attributes until you find the one you want to modify, and then click to select it.
   Once you have selected the attribute, the Name and OAT fields below will contain the current OAT setting.
5. Make your changes in the Name and OAT field and then click the **Set** button.
6. Click **OK** to dismiss the dialog box.

Deleting an OAT

Follow these steps to delete an OAT:

1. Navigate through the design to the component, component pin, or net that has the OAT you want to delete.
2. Double-click the component, component-pin, or net. The **Properties** dialog box appears.
3. Select the **Attribute** tab.
4. Scroll through the list of attributes until you find the one you want to delete, and then click on it to select it.
   Once you have selected the attribute the Name and OAT fields below will contain the current OAT setting.
5. Click the **Delete** button to remove the OAT.
6. Click **OK** to dismiss the dialog box
   If you have your OAT set to visible, you can also delete an OAT by clicking on the OAT in the schematic or symbol and then selecting the **Edit - Delete** command.

Reading and Writing OATs

All the information about OATs is written to a *projectname*.oat file. When you save a file, ViewDraw saves all the OATs information to a **.oat** file. ViewDraw reads in the OAT information from the **.oat** file when you open a file **(File - Open**). ViewDraw looks for an **.oat** file with the same name as the schematic you are opening.

ViewDraw creates and updates this **.oat** file when you perform a **File - Save** or **File - Save and Check** on a design with a manually added OAT. This file is also updated when you run the PCB Interface with OATs enabled.

An Important Note About OATS:

To use OATs, you should open the project at the top level and push through the levels of hierarchy. OATs are associated from the top–level schematic to the hierarchical level of the occurrence.

Making OATs Visible

You can set whether or not an OAT is visible on the design if you have a component, net, or component pin value that exists for that attribute. You can have the OAT name visible, the OAT value visible, the OAT name and value visible or you can have them all invisible. You can set this at the time you are adding the OAT by selecting the visibility option in the Attributes tab before setting up the attribute.

If you want to change the visibility of an existing OAT, you can do that in one of three ways:

If your OAT is already set to visible, you can click on the OAT and use the visibility toolbar buttons to change the setting.

-Or-

1. Double-click on the component, component-pin, or net.
   The Properties dialog box appears.
2. Select the **Attribute** tab.
3. Scroll through the list of attributes until you find the one you want to modify, and then click on it to select it.
4. Change the visibility setting.
5. Click on the **Set** button.
6. Click **OK** to dismiss the dialog box.

-Or-

1. You can select the OAT.
2. Select the **Edit -- Attribute Visibility** command.
3. Make the desired changes.
4. Click **OK** to dismiss the dialog box.

To select an invisible OAT:

1. Select Edit -- Select.
2. Click on the Attribute.
3. Type in the Attribute string (wildcards are supported).
4. Click Select.
5. Click Close.
6. Use the toolbar buttons to modify the visibility.

Copying and Moving OATs

Copying Oats

Since OATs are specific to hierarchical instanaces, you cannot copy OATs. When you copy an object that contains OATs using the **Edit - Copy** command, ViewDraw does not copy the associated OATs. Using the command, you can copy the selected objects, but not the associated OATs.

You can use the Copy/Paste feature in the Oatstools utility to copy OATs later once the design is saved. For more information on using this feature refer to the Propogate Tab topic .

Moving Oats

When you move a component (by dragging it across the schematic sheet), the associated OAT is retained. If you try to move the component by using the **Edit - Copy** and **Edit - Paste** commands (either across sheets on the same hierarchical level or at different hierarchical levels), the OATs are not retained since they are hierarchical path specific.

Getting OAT Reports

To get information about the OATs included in your design, select the **Project - Block - Oat** Details command. This command executes a summary of all the OATs in the design including the design hierarchy and the OAT locations.

The following is an small sample of the type of information generated when you use the **Project - Block - Oat Detail** command:

```
TOP LEVEL DESIGN: ser2par

SYMBOL: ser2par

PATH:

Sheet Number 1

$1I2 DELAY=1.25:1.55
```

Things to Consider When Using OATs

The following information will help you understand some of the issues associated with using OATs.

Top Level Considerations

Remember that when you use OATs, you should always open the project at the top level and push through the levels of hierarchy. OATs are associated from the top–level schematic to the hierarchical level of the occurrence.

Orphan OATs Considerations

Orphan OATs are Attributes left in the .oat file that is no longer associated with an object occurrence in the schematic. Orphans can be generated by any of the following editing processes:

- Relabeling Nets and Pins
  Changing a label in the hierarchical path to an OAT can be a problem because more than one net can share the same net label. When you delete a net, ViewDraw cannot determine if this is the only net with this particular label. Therefore, ViewDraw cannot safely delete any OATs at that time.
  After creating a net orphan, you can change the net labels to an existing label that had OATs associated with it. Because the OAT still exists, the net takes on the properties of that label (including the OAT). The risk is that these new OATs can be many revisions old and may be irrelevant to the current state of the project. Similarly for pins, changing the pin label orphans the corresponding OATs.
- Deleting Nets
  This leaves an unattached (orphaned) OAT behind. Deleted nets remove only the selected net segments from one sheet. You should delete the net OAT through the Properties dialog box before you delete the net.
- Editing Symbols
  In a hierarchical design environment, you can reference a single symbol many times in a project, as well as in multiple projects. You can change a symbol and the underlying schematic in one place, and this change automatically occurs in all projects that reference the symbol. Editing symbols can cause component or component pin orphans. The .oat file is not synchronized with the project because the .oat file is associated with the entire project, not the individual blocks of the project.
- Restoring Schematics
  Doing this might remove any object that has an OAT attached. This may happen because the **File - Restore** command operates on a single sheet and OATs are associated by project.

You can remove these orphans by running the Oatcheck utility. You must run Oatcheck before netlisting to PCB layout or simulation to insure a clean netlist. You should run the Oatcheck utility often when you are editing a schematic that contains OATs. Refer to the Oatcheck utility topic for more information about running Oatcheck.

Restore File Considerations

If you execute a file restore from the middle of the design, this will restore the design without restoring the OATs that are part of the schematic.

Copying Oats

Since they are specific to a particular hierarchy, you cannot copy OATs. When you copy an object that contains OATs using the **Edit - Copy** command, ViewDraw does not copy the associated OATs. Using the command, you can copy the selected objects, but not the associated OATs.

OATs FAQs

Here is a list of the most commonly asked questions about using OATs.

- How do I propogate my existing OATs files to a new top level?
- Why do my OATs with no default value show up differently?
- How do I enable OATs?
- How do I make the OATs visible?
- How do I copy OATs from one instance to another?
- How do I insert and delete sheets in my design?
- What if I did not select all relevent .oat files when inserting and deleting a sheet?
- How do I undo what Oattools did?

OATs APIs

API to OATs

You can access OATs functionality through the ViewDraw User Interface, or through an API. The APIs for OATs are available through automation and ViewBase.

Automation Interfaces for OATs

AddOat

AddBatchAttributes

GetEitherValue

SetEitherValue

GetOatValue

Value

ViewBase Interfaces for OATs

```
igcomoat -- Get first component OAT

ignetoat -- Get first net OAT
```

```
igocsatt  -- Get OAT, then component, then symbol
igonatt -- Get OAT, then net, then symbol
igopsatt -- Get OAT, then component pin, then symbol
igpinoat -- Get first component pin OAT
ifocsatt -- Find OAT, then component, then symbol
ifonatt -- Find OAT, then net, then symbol
ifopsatt -- Find OAT, then component pin, then symbol
iwloadoat -- Load a specified OAT file from a group
iwoat -- Load all OAT files for a group
```

For details on ViewBase Interfaces for OATs, refer to the ViewBase online help.

ViewDraw Reference

Shortcut Keys

The following shortcut keys (or key bindings) are available in ViewDraw:

PC Key Bindings

c or I	Add Component
n	Add Net
b	Add Bus
d	Delete
f	Add Fub
h	Push Schematic
r	redo (Ctrl A)
s	Select Mode (Same as <Esc>)
u	Undo (Ctrl Z)
w	Save and Check
y	Push Symbol
z	Zoom (F9)
A	Add Arc
B	Add Box
C	Add Circle
L	Add Line
T	Add Text
W	Save As
Ctrl C	Edit Copy
Ctrl-Del	Delete Special

Ctrl N	New File
Ctrl O	Open File
Ctrl P	Print
Ctrl r	Re-attach Net
Ctrl S	Save
Ctrl X	Edit Cut
Ctrl V	Edit Paste
Ctrl Z	Edit Undo
F1	Help
F2	Left Mouse Button
F3	Right Mouse Button
F4	View Full
F5	Refresh
F6	Pan
F7	View In
F8	View Out
F9	View Zoom
Delete	Delete
Shift Z	Zoom Select

ViewDraw Menus

ViewDraw has the following menus:

- File
- Edit
- View
- Add
- Project
- Tools
- Analog
- Window
- Help

Command Line Commands

A B C E F G H I L N O P

R S T U V W Z

A-

- a0size -- Change the sheet size.
- a1size -- Change the sheet size.
- a2size -- Change the sheet size.
- a3size -- Change the sheet size.
- a4size-- Change the sheet size.
- adistance-- Change the avoidance distance.
- ainvis-- Make the attribute invisible.
- anvis-- Make the attribute name visible.
- aoff-- Turn off attribute visibility.
- aon-- Turn on attribute visibility.
- arc-- Draw and arc.
- aroute-- Change to avoidance routing.
- array-- Create an array.
- arrowson -- Turn on pintype arrows
- arrowsoff -- Turn off pintype arrows
- asize-- Change the sheet size.
- attr-- Add attribute text.
- avis-- Make the attribute visible.
- avvis-- Make the attribute value visible.

-B-

- bannotate-- Change to annotation block.
- bb -- Modify bounding box
- bcomposite-- Change to composite block
- bcopy-- Copy to the buffer.
- bcut-- Cut to the buffer.
- bmodule-- Change to module block.
- boff -- Turn off the border
- bon-- Turn on the border
- box-- Draw a box

- bpaste-- Paste from the buffer
- bpin-- Change to a pin block.
- bsize-- Change the sheet size.
- bus-- Draw a bus.

-C-

- cattribute-- Change properties for all attributes.
- ccomp-- Change component
- chgattr-- Change properties for selected attribute
- chglabel-- Change selected label.
- chgtext-- Change selected text
- cinst-- Update pin numbers
- circle-- Draw a circle.
- component -- Add a component.
- color-- Change color
- copy-- Copy object
- csize-- Changes the sheet size.
- ctoff-- Toggle text off
- cton -- Toggle text on.

-D-

- dbevoff-- Turn off verbose error.
- dbevon-- Turn on verbose error.
- delete-- Delete object.
- directory-- List schematics and symbols
- dirsch-- List schematics
- dirsym-- List symbols
- doff-- Toggle continuous object display off
- don-- Toggle continuous object display on.
- dsize-- Change the sheet size.

-E-

- esize-- Change the sheet size.

-F-

- full-- Show full view.

-G-

- global-- Change to global scope.
- goff-- Toggle grid off.
- gon-- Toggle grid on.
- gspace-- Change grid spacing

-H-

- hoff-- Toggle header off.
- hon-- Toggle header on

-I-

- in -- Zoom in.

-L-

- label -- Add a label.
- line -- Draw a line.
- linvis-- Make label invisible
- loff-- Turn off display of all labels.
- local-- Change to local scope.
- lon -- Turn on display of all labels.
- longstrings -- Search for strings greater than 127 characters.
- lsense-- Invert label sense.
- lvis-- Make label visible.

-N-

- nainv-- Make new attributes invisible
- nanv-- Make new attribute names visible.
- navis-- Make new attributes visible.
- navvis-- Make new attribute values visible.
- name-- Change the attribute name
- net -- Draw a net.
- noff-- Toggle display of internal names off.
- non-- Toggle display of internal names on.

-O-

- o1 -- Set the origin point for text.
- o2 -- Set the origin point for text.

- o3 -- Set the origin point for text.
- o4 -- Set the origin point for text.
- o5 -- Set the origin point for text.
- o6 -- Set the origin point for text.
- o7 -- Set the origin point for text.
- o8 -- Set the origin point for text.
- o9 -- Set the origin point for text.
- oatsdetail -- Give information on OATs.
- oattribute -- Give information on object attributes.
- odetail-- Give information of objects.
- oroute-- Change to orthogonal routing
- out-- Zoom out.

-P-

- pin -- Draw a pin.
- poff-- Turn off pin numbers
- pon -- Turn on pin numbers.
- Pop -- Move up in the hierarchy.
- promote-- Promote symbol attributes
- psch -- Push to the underlying schematic.
- Psh -- Push to the next sheet.
- psense-- Toggle the sense of a pin.
- psheet-- Push to a given sheet.
- psym-- Push to the corresponding symbol for a schematic.

-R-

- reflect-- Reflect an object around an axis.
- refresh-- Refresh the screen.
- read-- Refresh memory with the active block.
- reread-- Reread all data in memory.
- roff-- Turn off display of REFDES
- ron-- Turn on display of REFDES
- rotate-- Rotate an object.
- run -- Run a script.

-S-

- sattr-- Select attribute
- save-- Write to disk
- savepin-- Write PINORDER to file.
- scale-- Change object size
- schematic -- Open a schematic.
- scomp-- Select component
- sdistance-- Define selection distance
- size-- Change size of textual data
- slabel-- Select label
- slot-- Change component slot.
- sname-- Select object by name.
- soff-- Turn off snap-to-pin
- son-- Turn on snap-to-pin.
- sortoff-- Turn off alphabetic name sorting.
- sorton-- Turn on alphabetic name sorting.
- sroute-- Change to straight routing
- stext-- Select text string.
- stretch-- Stretch object.
- string-- Change string.
- svalue-- Select object by value
- symbol-- Open a symbol

-T-

- text-- Add text.
- threshold-- Change text display threshold.
- toff-- Turn off text visibility.
- ton-- Turn on text visibility.

-U-

- undo-- Undo change.
- uoff-- Turn off unique labeling
- uon-- Turn on unique labeling.

-V-

- value-- Change attribute value.

- voff-- Turn off simulation value visibility
- von-- Turn on simulation value visibility.

-W-

- wclose-- Close active drawing.
- write-- Save and check drawing.
- writinit -- Write to viewdraw.ini
- wte--Open text editor.

-Z-

- zoom-- Zoom in on specified section.
- zselect-- Zoom in on selected section.
- zsize-- Change the sheet size.

Dialog Boxes

The following dialog boxes appear as part of the ViewDraw application:

- About Box
- Add Component
- Add Fub
- Find/Replace
- File Open
- Insert Object
- Modify Attribute Visibility
- Paste Special
- Print
- Print to File
- Properties
- Project Settings
- Replace
- Save As...
- Scale
- Select
- Slot
- Write
- Color Palette

ViewDraw FAQs

Here is a list of commonly asked questions about using ViewDraw

Design Entry Tips:

- How do I draw diagonal nets?
- Does Viewdraw have a net disconnect hotkey?
- When using OATS, how can I tell which occurrence I am looking at?

Netlisting:

- How are alphanumeric bus labels treated by the Netlister?
- How can I customize full version ViewDraw's Tools Menu to create an EDIF Netlist for Actel?

Correcting Errors:

- I am Getting Error 8037. How can I Correct this Error?
- I get error 6092 when I run PCBFWD or PCBBACK. How can I correct this error?
- Back Annotation from Layout is Giving me Errors
- Why am I getting a ViewDraw VLLIB error in Libero IDE?
- Why am I getting aVdconfig.exe errors when installing Libero IDE 5.0 on Windows98 or WindowsNT?

ViewDraw Utilities

Viewdraw Utilities Description

ViewDraw provides you with several types of utilities to assist you with your design flow.

Design Navigation:

ViewNavigator

ViewNavigator l locates objects in a ViewDraw Design, displays the design by hierarchy or REFDES, displays pins and nets, and navigates to errors.

For details, refer to the ViewNavigator online help.

Design Verification:

Schematic Checker

The Schematic Checker checks the specified sheet or project for minor connectivity violations and creates a new wir file.

For details, refer to the Schematic Checker online help.

Design Rule Checker

The Design Rule Checker locates electrical rule violations in your design and performs simple checks for the presence of information.

For details, refer to the Design Rule Checker online help.

Troubleshooting

Glossary

A

absolute spacing: Measures the column and row spacing from the lower left corner of the minimum bounding box that surrounds the selected object or group of objects.

add mode: Allows you to add multiple objects without having to invoke the Add object command each time. Enter this mode by selecting the Add command or object toolbar button while depressing the Ctrl key.

annotation file: A text file that is accessed from the current directory.

attached attributes: Attributes that apply to a particular pin, net segment, or component.

attributes: Text that is attached to schematic or symbol objects that provides additional information about the design (such as rise/fall time, delay, capacitance, resistance, etc.). The system interprets the attributes during the wirelist process.

avoidance routing mode: An automatic connection between two points that avoids components. Other routingmodes include Orthogonal and Straight.

B

block: A schematic or symbol sheet. A block has; a file name, a sheet number (1 to 999), a data type (schematic or symbol).

bus: A collection of nets that can operate as a group or as individual nets within the bus. A bus is created between components, from a single component, or between nets.

bus rippers: A bus ripper is a symbol that you instantiate in your design to pull signals from a bus. The ripper symbol provides an object between the bus and the net(s) being extracted from the bus. This provides you with an unambiguous reference to selected signals of the bus. You can also use bus rippers to create an alias netname.

C

component: An instantiation (or reference) of a symbol on a schematic drawing. Components allow you to use a symbol many times.

component array: A component array is a single component on a schematic that represents multiple components in the logical database.

composite block: You define a composite block with a corresponding internal schematic or wirelist. Composite blocks signify the hierarchy within the block.

D

dangling net: A net connected at only one end to a component. Dangling nets are referenced to all other net connections through the net label.

F

fub: An automated way to create a shell for a new symbol. When you create the symbol shell and add nets and buses to it, ViewDraw automatically updates the symbol to include the pins. When you save the symbol, ViewDraw labels the pins so that they correspond to the nets or buses they connect to. Fubs have the SYMTYPE=FUB attribute associated with it. You must remove this attribute and add other attributes before using the symbol created by the fub.

H

hierarchical path: A hierarchical path is used to define a specific occurrence of an object. This path consists of unique identifier for each object in the path, beginning with the top level component all the way down to an object on a lower-level schematic. This path also includes the internal name (or label) for the object at the end of the path. You form the unique hierarchical path - the occurrence - by selecting a component, pushing down to a lower level in the design hierarchy, and selecting an object on the component's underlying schematic. Example: $1I34\$1I27\$2I14

I

instance: An instance is any component, component pin, or net on a schematic sheet.

L

label: A text label that uniquely identifies components, nets, pins, or buses. Labels uniquely identify objects for wirelisting and ViewSim. Components and pins can have only one label. Nets and buses can have multiple labels of the same name -- one label per net or bus segment. A label can express a numerical range for a bus or pin by specifying beginning, ending, and incremental integer values. A label with a range is expanded to create a series of labels. The format of the label is name[F:L:I] where F is the first number, L is the last number, and I is the interval between the numbers in the range. If no interval is specified, the interval is 1. (A radix qualifier can follow the range specification to indicate a binary, hexadecimal, or octal specification.)

label range: A label can express a numerical range for a bus or pin by specifying beginning, ending, and incremental integer values. A label with a range is expanded to create a series of labels. The format of the label is name[F:L:I] where F is the first number, L is the last number, and I is the interval between the numbers in the range. If no interval is specified, the interval is 1. (A radix qualifier can follow the range specification to indicate a binary, hexadecimal, or octal specification.)

label scope: Label scopes can be global or local. Global labels are known throughout the hierarchy. Flattened wirelists interpret global labels. However, hierarchical wirelist interpretation depends on the capability of the target system (for example, HSPICE supports global labels). Local labels are only known within the schematic at one level in the hierarchy.

line: A line connects two or more points. The portion of a line between two points is a line segment. Lines do not represent connectivity.

M

module block: A symbol with no corresponding internal schematic. Another name for a module block is a primitive. Use module blocks to create flat designs.

N

net: Represents an electrical connection. You create a net between components, from a single component, or between nets. A net is constructed with one or more segments. If a net has more than one segment, the segment endpoints are indicated by joints at the net vertices. You can connect a maximum of four nets at a net joint.

O

occurrence: There is only one path from a component on the top-level schematic to a selected instance on a lower-level schematic. This unique hierarchical path, along with the selected instance's ID, constitutes an occurrence.An occurrence differs from an instance in that an occurrence consists of a unique hierarchical path to a specific instance within the design hierarchy (whereas an instance can have many paths to it). If you use a component only once in a design, then the component instance and component occurrence are the same. You form the unique hierarchical path - the occurrence - by selecting a component, pushing down to a lower level in the design hierarchy, and selecting an object on the component's underlying schematic.

orthogonal routing mode: A 90 degree connection between two points. Other routing modes include Avoidance and Straight.

P

pass-through pins: Two or more pins on a symbol with the same label. You typically place these pins on opposite sides of the symbol. Designs using pass-through pins must be flattened before they are sent to a downstream simulator. Do not use pass-through pins for hierarchical netlisting, such as Export1076; or if the pins exist at the MODULE level, such as Zycad, C models, Verilog, LMC models, VHDL, or compiled code.

pin: A pin provides a connection point for nets that carry signals into or out of a component.

pin block: The file and sheet used to represents a schematic pin.

project: The collection of information used to build your schematic or symbol. The information is stored in subdirectories under the main project directory. ViewDraw uses projects to manage designs and libraries.

R

relative spacing: Measures the column and row spacing from the border of the minimum bounding box that surrounds the selected object or group of objects.

routing mode: All connectivity is formed by specifying a routing mode. Valid routing modes are: straight, avoidance, or orthogonal.

S

scale factor: The scale factor is the amount you choose to reduce or enlarge an object. Determine this factor using this formula: scale factor = (scaled size) / (current size) .

search order: The search order specifies libraries available to a particular design.

sheet: The equivalent of a page of the drawing. To avoid crowding the design, spread out the design on consecutive sheets. These consecutive sheets are called the sheet set of a drawing. The system references sheet numbers by the schematic filename extension. For example, DMA.1 refers to sheet 1 of schematic DMA; LOWPASS.4 refers to sheet 4 of schematic LOWPASS.

sheet border: A graphic that frames the schematic or symbol.

straight routing mode: A straight connection between two points. Other routing modes include avoidance and orthogonal.

symbol: A graphical representation of a block type on a schematic sheet. Symbols are written to the sym subdirectory.

symbol block: A border for creating a symbol. When you add a pin to the symbol body, the pin snaps to the nearest edge of the border.

U

unattached attributes: Attributes that apply to a symbol or schematic in general.

Unique Labels: ViewDraw creates different types of labels when components, nets, buses, or pins are copied, arrayed, or pasted from the buffer. Each label type is discussed in this section.

V

vectorized labels: Vectorized labels are a shorthand notation for repetitive labeling.

viewer screen: A window located at the bottom right corner of the Add Component dialog box that displays a view of the selected component. You can drag components from this screen and drop them in place on the schematic sheet.

Index

For more information about Actel’s products, call 888-99-ACTEL or visit our Web site at http://www.actel.com

Actel Corporation • 2061 Stierlin Court • Mountain View, CA USA 94043
U.S. Toll Free Line: 888-99-ACTEL • Customer Service: 650-318-4200 • Customer Service FAX: 650-318-2440
Customer Applications Center: 800.262.1060 • Customer Applications FAX: 650.318.4600

Actel Europe Ltd. • Dunlop House, Riverside Way • Camberley, Surrey GU15 3YL • United Kingdom
Tel: +44 (0) 1276 401 450 • Fax: +44 (0) 1276 401 490

Actel Japan • EXOS Ebisu Bldg. 4F • 1-24-14 Ebisu Shibuya-ku • Tokyo 150 • Japan
Tel: +81.03.3445.7671 • Fax: +81.03.3445.7668

Actel Hong Kong International • Suite 2114 • Two Pacific Place • 88 Queensway, Admiralty • Hong Kong
Tel: +852 2185 6460 • Fax: +852 2185 6488