Allegro Design Entry HDL Reference Guide

Product Version 23.1
September 2023

Cadence Design Systems, Inc. (Cadence), 2655 Seely Ave., San Jose, CA 95134, USA.

Disclaimer: Information in this publication is subject to change without notice and does not represent a commitment on the part of Cadence. Except as may be explicitly set forth in such agreement, Cadence does not make, and expressly disclaims, any representations or warranties as to the completeness, accuracy or usefulness of the information contained in this document. Cadence does not warrant that use of such information will not infringe any third party rights, nor does Cadence assume any liability for damages or costs of any kind that may result from use of such information. Cadence is committed to using respectful language in our code and communications. We are also active in the removal and/or replacement of inappropriate language from existing content. This product documentation may however contain material that is no longer considered appropriate but still reflects long-standing industry terminology. Such content will be addressed at a time when the related software can be updated without end-user impact.

Restricted Rights: Use, duplication, or disclosure by the Government is subject to restrictions as set forth in FAR52.227-14 and DFAR252.227-7013 et seq. or its successor.

Contents

Preface

About Allegro Design Entry HDL Reference Guide

The *Allegro® Design Entry HDL Reference Guide* provides reference information, such as dialog box and menu help to assist you in using the Allegro® Design Entry HDL schematic editor.

This reference guide assumes that you are familiar with the development and design of electronic circuits at the system or board level.

Finding Information in This User Guide

This user guide covers the following topics:

See...	**For Information About...**
Appendix 1, "Console Command Reference,"	Describes the Design Entry HDL console commands
Appendix 2, "Nongraphical Design Entry HDL (nconcepthdl),"	Describes how to use Design Entry HDL in batch mode
Appendix 3, "Using the Standard Library Symbols,"	Describes the components available in the Design Entry HDL Standard library
Appendix 4, "Error Checking Features in Design Entry HDL,"	Error checking features in Design Entry HDL and description of some netlisting errors
Appendix 5, "Design Entry HDL Files,"	Describes the various Design Entry HDL files
Appendix 6, "The template.tsg File,"	Describes how to create a template.tsg file
Appendix 8, "Dialog Box Help,"	Describes the various Design Entry HDL dialog boxes
Appendix 7, "Menu Help,"	Describes the various Design Entry HDL menus

See...	For Information About...
Appendix 9, “Naming Rules and Conventions,”	Specifies naming rules and conventions used in Design Entry HDL and related tools.
Appendix 11, “Managing Environment Variables,”	Describes the environment variables required to run Design Entry HDL and related tools and how to set these variables
Appendix 12, “Troubleshooting Allegro Design Entry HDL,”	Lists the solutions to some common or intermittent problems encountered while using Allegro Design Entry HDL.
Appendix 13, “Glossary,”	The Design Entry HDL glossary

Related Documentation

You can also refer the following documentation to know more about related tools and methodologies:

Design Entry HDL

- For information on the new features in 16.01, see *Allegro Design Entry HDL: What’s New in Release 16.01*.
- For learning Design Entry HDL, see *Allegro Design Entry HDL Tutorial*.
- For learning to use Constraint Manager with Design Entry HDL, see *Allegro Constraint Manager with Design Entry HDL Tutorial*.
- For information about the SKILL interface to Design Entry HDL, see *Allegro Design Entry HDL SKILL Reference.*

Front-to-Back Flow

- For information on the front-to-back flow for PCB design, see *Allegro Front-to-Back User Guide*.
- For information on the Design Synchronization solution, see *Design Synchronization and Packaging User Guide* and *Design Synchr(onization Tutorial*.
- For information about packaging your design, see the *Packager-XL Reference*.

- For information on Design Variance solution, see the *Design Variance User Guide* and *Design Variance Tutorial*.

Related Tools and Flows

- For information on various PCB design working environments such as a team of designers working on a Design Entry HDL project, implementing FPGAs in designs, working with high-speed constraints, importing IFF files for radio-frequency designs, and reusing existing modules, see *Allegro PCB Design Flows*.
- For learning how to create new Design Entry HDL projects and make various settings for them, see the *Allegro Project Manager User Guide*.
- For learning how to use the Design Entry HDL utilities - CRefer, Archiver and BOM, see the Allegro *Design Entry HDL Utilities User Guide*.
- For information on maintaining and modifying the Design Entry HDL digital libraries, see the *Allegro PCB Librarian Expert User Guide*, *Part Developer User Guide*, and *Allegro Design Entry HDL Libraries Reference*.
- For information on the digital simulation interface provided by Design Entry HDL, see *Allegro Design Entry HDL Digital Simulation User Guide* and *Allegro Design Entry HDL Digital Simulation Tutorial*.
- For learning the Design Entry HDL Programmable IC flow, see *Programmable IC Tutorial*.
- For information on capturing electrical constraints in Constraint Manager, see the *Allegro Constraint Manager User Guide*.
- For information on Design Entry HDL Rules Checker, see *Allegro Design Entry HDL Rules Checker User Guide*.
- For information about moving from SCALD to HDL flow, see *SCALD to HDL Evolution Guide*.
- For information on creating custom interfaces to translate the HDL database into a format that can be used by an external system and to update the HDL database with changes from a physical design system, see the *CAE Views Programming Guide*.

Typographic and Syntax Conventions

This list describes the syntax conventions used for this user guide:

`literal`	Nonitalic words indicate keywords that you must enter literally. These keywords represent command (function, routine) or option names.
`argument`	Words in italics indicate user-defined arguments for which you must substitute a name or a value.
|	Vertical bars (OR-bars) separate possible choices for a single argument. They take precedence over any other character.
`[ ]`	Brackets denote optional arguments. When used with OR-bars, they enclose a list of choices. You can choose one argument from the list.
`{ }`	Braces are used with OR-bars and enclose a list of choices. You must choose one argument from the list.

1

Console Command Reference

You use console commands to perform all operations in Design Entry HDL which you can perform using menu commands or toolbar icons. This section describes the commands that you can enter in the Design Entry HDL console window. The syntax, abbreviation, description, and related commands for console commands are described below.

Add

Syntax Procedure Dialog Boxes Related Commands

Syntax

```
ADd [component_name][.view][.version] [point] [point...] | <cr>
```

This command adds a specified component to a drawing. Component_name is the name of the component to be added. View is the symbol view. The version is 1 by default, but any existing version of a component can be added.

To add a component to a schematic, type `ADD` <*component_name*>. A copy of the component is attached to the cursor. Press the right mouse button and select *Version* to cycle through different versions of the component. Move the component to the required position in the drawing and click. To add another copy of the component, click, then position the copy as required.

ADD <cr> accesses the Part Information Manager dialog box. You can use this dialog box to add components to the drawing. The Part Information Manager dialog box can be turned off by typing in SET ADDFORM OFF.

Use the REPLACE command to substitute one component for another.

Dialog Boxes

Add Component-Library View

Add Component-Category View

Related Commands

Rotate

Replace

PPTAdd

Version

Arc

Syntax Procedure Related Commands

Syntax

```
ARc point1 point2 { point3 | ; }
```

Description

This command creates arcs, usually on symbols. The two points define the ends of the arc. The curvature of the arc is controlled dynamically by dragging the mouse after you place the second point.

Click to place the arc at the nearest screen pixel. Click the center button to place the arc at the nearest grid intersection (useful when building accurate semicircles or matching mirrored arcs).

Typing a semi-colon after placing the second point will create a circle.

Related Commands

Circle

Assign

Syntax

```
Assign function_key "quoted-string"
```

Description

This command assigns an editor command or operation to a programmable function key. You can press the specified key instead of typing the text. This saves time when a command is used often or requires several variables and options on the command line.

To assign a string to a key, enter the key name or press the function key and then type in the command text to be assigned to the key. Enclose the command and its arguments in quotation marks. They can be uppercase or lowercase. Note that the shift and control keys can also be used, thus allowing up to three different assignments to each function key.

For example,

```
ASSIGN F2 "zoom fit"
ASSIGN <shift>F2 "zoom in"
ASSIGN <cntrl>F2 "zoom out"
```

Function key names correspond as closely as possible with the text printed on the keyboard. The function keys for the various systems are:

IBM:	F6-F10, F12, 4-9, page up/down, the directional arrows
Sun:	F1-F9, R4-R12

SHOW KEYS lists the current function key assignments. ECHO <key press> displays the key name or the assignment.

Attribute

Procedures Dialog Box

Syntax

```
Attribute point
```

Description

This command accesses the Attribute dialog box for adding, modifying, or deleting properties on a drawing. To use the command, type ATTRIBUTE, point to the desired object, and click the left mouse button. The editor brings up a form containing all the properties attached to the selected object.

Procedures

- Adding Properties
- Displaying and Modifying Property Attributes
- Making an Attribute File

Auto

Syntax

```
AUto {PAth | Dots | OccProperty | Undot | PRoperty group_name prop_name prop_value
. . }
```

Description

This command performs the global addition or deletion of certain objects to or from a drawing.

- PATH automatically assigns unique PATH numbers to bodies on a drawing that do not already have a PATH property. Some special editor bodies and bodies labeled with a COMMENT property are not assigned PATH properties. The path property is in the form 'PATH = In', where `n` is a unique integer.
- DOTS places a dot at each wire connection on the current drawing. Open dots are the default value. To specify filled dots, type SET DOTS_FILLED before entering AUTO DOTS.
- OCCPROPERTY adds user properties onto groups of components and allows you to specify the property in the context of the root design. Type `auto occproperty` *`<group_name> <property_name>=<property_value>`*

  This will attach the property with the value that you specify in *`<property_name>`* *`<property_value>`* to the specified group of objects, *`<group_name>`*, and set the source to the root design.

  Be sure to edit the design in the right context before executing this command.

- UNDOT removes all dots from the drawing except those at the intersection of four wires.
- PROPERTY adds user properties onto groups of components.

  Note: By default, this command sets the source of the design to the block level. If you want to specify the root design as the source, use the `auto occproperty` command.
- GROUP_NAME is either a single letter identifying a group or a mouse click specifying the nearest group. Any number of property name-value pairs may be specified after the group, and the names and values may be separated by spaces, an equal sign, or a new line. Using this option will cause the entered properties to be automatically annotated onto the components within the group.

See also the SET DOTS_FILLED and SET DOTS_OPEN commands.

Auto Commands

Using the auto dot command

You can use the auto dot command to place dots on a complex circuit. The auto dot option with the `set` command automatically places dots on a drawing as you are creating it (`set autodot on`). Automatic dotting places dots at all intersections with an odd number of wires.

1. When `set autodot` is `off`, type

   ```
   show connections
   ```

   This command places asterisks temporarily on the drawing to highlight each connection point.
2. Check the drawing to make sure that no connections have been made by mistake.
3. Use the refresh command to remove the asterisks from the screen.
4. Type

   ```
   auto dot
   ```

   All the junctions are automatically dotted.

Using the auto undot command

To remove the dots of the same size as the present dot size settings at intersections in a drawing type

```
auto undot
```

Note: You cannot use the `auto dot` and `auto undot` commands separated by a semi colon (;) on the same line in the console window. The following usage results in an error:

```
auto dot; autoundot
```

Using the auto allundot command

To remove all the dots at the intersections in a drawing type

```
auto allundot
```

Changing Dot Size

- If you change the size of a dot, the change is not reflected on filled dots in the schematic, however, when you plot the schematic, the change in size gets reflected.
- Change in size is also not reflected in dots that existed previously, it appears only in the dots that you add after changing the size. So, if you want to make all the dots of the same size, first use `auto undot` to remove the dots within the size specified in logic dot radius, or `auto allundot` to remove all dots regardless of size. Then add the dots again using `auto dot`.

Using the auto netprop command

Use this command to apply an attribute list to a set of wires. To attach the specified property on all wires in a group, type

```
auto netprop <group_name> <property_name> = <property_value>
```

Using the auto path command

If the `SET` command option `AUTOPATH` is on, the PATH property is automatically added to a part when it is added to a drawing. If `set autopath` is off, you can use the `auto path` command to assign PATH properties to symbols that do not already have a PATH property in a drawing.

Using the auto property command

To attach properties to symbols in a group, type

```
auto property <group_name> <property_name> = <property_value>
```

Using the autoroute command

The `set` command option `autoroute` on activates automatic routing after moving an object in the direct mode. The set autoroute option can be turned on or off.

Backannotate

[Syntax](#) [Procedure](#)

Syntax

```
Backannotate {annotation file | <cr>}
```

Description

This command annotates designs with physical information from the Packager. The editor reads the specified schematic annotation file produced by the Packager. The file includes physical information such as location designators, pin numbers, physical net names on the design, and user-defined properties, if any.

The annotated properties added by the editor are soft properties. Soft property names begin with a dollar sign (for example, $LOCATION) and are not written into the connectivity file. This allows Packager to reassign physical information each time the design is repackaged.

You can move and delete soft properties, or you can change a soft property into a hard property by using the PROPERTY command and adding a property with the same property name without the dollar sign.

To generate a backannotation file, use the following directive when running the Packager:

```
output backannotation;
```

By default, this directive is set.

To process the backannotation file generated by the Packager (`pstback.dat`), type either of the commands BACKANNOTATE PSTBACK.DAT or BACKANNOTATE <cr>. BACKANNOTATE <cr> brings up the File Browser form. The user can then select the name of the backannotation file from the form. The editor reads the file, edits each named drawing in turn, adds the appropriate physical information, and writes the drawing. Backannotation aborts if any errors are detected during the process.

Caution

Do not run backannotation if any other user who has write permissions is working on the design. Running backannotation when another user is working on the design results in incomplete backannotation.

Limitations of Backannotation

1. Sizeable parts with a property `SIZE > 1` are not backannotated.
2. Net properties are backannotated only if they have existing placeholders.
3. Properties on buses are not backannotated unless they exist on individual bits of the bus.

See also the SET command options that control property visibility and pin number placement.

Badd

Syntax Procedure Related Commands

Syntax

```
BADd { [block_name] point1 point2 } . . .
```

Description

This command can be used to create and add blocks. The block is a rectangle between point1 and point2. If you omit `block_name`, Design Entry HDL automatically names the block (the name will be "BLOCK" followed by an integer). If you enter a `block_name`, and a symbol for that part already exists, the add command is displayed.

Example

To add a new block `CACHE` to the design, do the following:

1. Type `badd CACHE` in the console window.
2. Click the design window and drag to form a rectangle.

   Note: The cursor shape changes to a + sign.

3. Click the design window again to end the drawing.

Related Commands

Add
Bpadd
Bpdelete
Bprename
Bpmove
Brename
Broute
Bstretch
Bwire

Bpadd

Syntax Procedure Related Commands

Syntax

```
BPAdd {pin_name :mode point}...
```

Description

Add or rename pins on blocks. bpadd adds pins to blocks with a user-specified name. It is useful if you want to define the interface pins of a block before you connect up your blocks. mode specifies whether the block pin is an input pin, an output pin, or an inout pin, and can take values `input`, `output`, `inout`, respectively. You can change the mode of the pin while you are placing it on the block. pin_name is the name of the pin that will be created. Point is where the pin is created.

Related Commands

Badd
Bpdelete
Bprename
Bpmove
Brename
Broute

Bstretch
Bwire

Bindview

Syntax

```
Bindview point1 point2
```

Description

This command makes a drawing that is visible in one viewport visible in another viewport also. Any changes you make to one copy of the drawing also appear on the other copy.

Point1 selects the drawing to make visible. Point2 specifies the new viewport where the drawing should appear.

When you use BINDVIEW, the drawing name is not added to the drawing stack in the second viewport. If you issue a command, such as SHOW or RETURN, the bound drawing does not appear in the list. When you EDIT or RETURN to a drawing in the bound viewport, the binding is removed. Use BINDVIEW again to re-bind the drawing.

Bpdelete

Syntax Procedure Related Commands

Syntax

```
BPDelete {point}...
```

Description

This command deletes pins on blocks. You can choose the pins to be deleted by pointing to them with the mouse.

Example

To delete a pin, type `bpdelete` in the console window and select the pin using the left mouse button.

Related Commands

Badd
Bpadd
Bprename
Bpmove
Brename
Broute
Bstretch
Bwire

Bpmove

Syntax Procedure Related Commands

Syntax

```
BPMove {point1 point2}...
```

Description

This command moves a pin from point1 on a block to point2.

Example

To move a pin from point1 on a block to point2:

1. Type `bpmove` in the console window.
2. Click to select the pin to be moved (point1).
3. Click to place the pin on the block (point2).

If you try to undo this operation, some hanging properties may remain on the schematic. This is because the `bpmove` command actually moves a pin and its properties on the symbol while the `undo` command operates only on the schematic.

To remove these hanging properties, perform the following steps:

1. `set sticky_on`
2. `get`
3. `set sticky_off`

Related Commands

Badd
Bpadd
Bpdelete
Bprename
Brename
Broute
Bstretch
Bwire

Bprename

Syntax Procedure Dialog Box Related Commands

Syntax

```
BPRename {pin_name point}...
```

Description

This command renames pins on blocks. In this command, pin_name is the new name of the pin and point is location of the pin.

Example

To rename a pin named `PRESET` to `CLOCK`, type `bprename CLOCK` in the console window and click *PRESET*.

Related Commands

Badd
Bpadd
Bpdelete
Bpmove
Brename
Broute
Bstretch
Bwire

Brename

Syntax Procedure Dialog Box Related Commands

Syntax

```
BREname {[block_name] point1}...
```

Description

This command renames blocks. This is useful if you allowed the system to generate default names for your blocks and want to rename them. It does not remove old blocks from the disk. It does, however, remove the old block from the memory if you are not actually viewing it. So if the old block had not been saved to disk, it will have been removed. If the old block had been saved to the disk, Design Entry HDL generates the following message:

```
You might want to remove <old_block_name> with the remove command.
```

If you do not need the old block any longer, you can remove it from the disk by using the `remove` command, for example `remove <old_block_name>`.

This command is also useful if you want to create a block similar to one you already have. You can copy the block (with the COPY command) and rename the copy of the block. The original block will be unaffected.

Example

To rename a block named `CACHE` to `MEMORY`, type `brename MEMORY` in the console window and click *CACHE*.

Related Commands

Badd
Bpadd
Bpdelete
Bpmove
Bprename
Broute
Bstretch
Bwire
Copy
Remove

Broute

Syntax Procedure Related Commands

Syntax

```
BROUte {[signal_name] point1 point2}...
```

Description

This command routes a signal named signal_name between point1 and point2. If the signal_name is a bus name, create a heavy wire.

Related Commands

Badd
Bpadd
Bpdelete
Bpmove
Bprename
Brename
Bstretch
Bwire
Route
Wire

Browse

Syntax

```
Browse {<libname> | <cr>}
```

Description

This command accesses the Part Information Manager dialog box. It lets you scan through all libraries and active directories. You can also access the Part Information Manager with the ADD command.

BROWSE <*`libname`*> sets the library field to the given library name and resets the selection list to that library. The default is the design library.

Example

To display the Part Information Manager dialog box, type `browse` in the console window and press `Enter`.

To display the Part Information Manager dialog box with the library set to *standard*, type `browse standard` in the console window and press `Enter`.

Bstretch

Syntax   Procedure   Related Commands

Syntax

```
BStretch {point1 point2} . . .
```

Description

This command resizes blocks. Select the side or corner of the block you want to stretch with point1. Then select where it stretches to with point2. Pins will move with the side they are attached to. Pins pointing to the left or right will only move horizontally. Pins pointing to the top or bottom only move vertically. You may not shrink a block so much that its pins or its origin fall outside it.

Wires attached to the instance of the block being stretched will be attached to new pin locations. However, they are not re-routed. If a new wire stub is not straight, use the SPLIT and DELETE commands to straighten it.

Example

To resize a block, type `bstretch` in the console window and select the side or corner of the block you want to stretch. Drag to the point you want to resize the block. Click again to end the block.

Related Commands

Badd
Bpadd
Bpdelete
Bpmove
Bprename
Brename
Broute
Bwire
Delete
Split

Bubble

Syntax Procedure

Syntax

```
BUBble point...
```

Description

This command toggles the state of a pin between bubbled and unbubbled. Bodies must be defined with bubbleable pins to permit this conversion. If the pins are established as part of a bubble group, the BUBBLE command can be used to convert the symbol from one form to another.

The `BUBBLED` property is automatically attached to bubbled pins to indicate that only low-asserted signals may be connected to them. The `BUBBLED` property should never be entered, assigned, or attached by the user.

You can also specify the coordinates of the pin in the console window instead of pointing to the pin in the design window. This is true for all console commands that require pointing to an object in the design window.

For example, a NOT symbol is defined with both the BUBBLED and BUBBLE_GROUP properties attached:

BUBBLED=(B)

BUBBLE_GROUP=(A|B)

Because BUBBLED is equal to (B), pin 'B' is bubbled when the part is initially added to a drawing. If you type BUBBLE and point to either pin A or B, the attached BUBBLE_GROUP property specifies that pin A is now the bubbled pin and pin B the unbubbled pin.

Example

To toggle the state of a pin, type `bubble` in the console window and click the pin in the design window.

To make two pins as part of the same bubble-group such that the BUBBLE command can be applied to the pins, you must ensure the following:

1. Both the pins must have a wire stub and a bubble attached to them, similar to the symbol of LS04 in the lstll library.
2. BUBBLE_GROUP property must be attached to the component symbol with the value containing all pin names in the bubble group. For example, BUBBLE_GROUP = (A<7..0> | B<7..0>).
3. The pin that you want to make bubbled by default, should be specified with a BUBBLED property on the component. For example, BUBBLED = B<7..0>

   Note: If the BUBBLED property is not specified, pins A and B appear as unbubbled on instantiation. When you run the BUBBLE command, both the pins will appear as bubbled. However, if the BUBBLED property has been specified only for pin B, then it will appear as bubbled by default, and pin A will appear as unbubbled. When you run the BUBBLE command again the A will appear as bubbled and B as unbubbled. Thus, the BUBBLE command acts as a toggle for changing the bubbled state of a pin.

Busname

Syntax Procedure Related Commands

Syntax

```
BUSName bus_name point point
```

Description

This command places single-bit vectored signal and pin names on a drawing. For example, the bus name A<7..0:2> results in the signal names A<7>, A<5>, A<3>, and A<1>.

To use the BUSNAME command, first place separate wires or pins (use the COPY command). Next, enter the bus name in Design Entry HDL signal syntax (for example, DATA<15..0>\I). Then, select two points. Design Entry HDL will find all the wires you have crossed and add the properties to those wires.

For examples, if you issue the following command:

```
busname a<20..0:2>\I
```

then pick two points, such that you have crossed over three wires, the crossed wires will be named as `a<20>\I`, `a<18>\I` and `a<16>\I` respectively, then you can select another two points and the wires crossed will be named `a<14>\I` and `a<14>\I` and so on.

If the first thing you do after entering the BUSNAME command is to enter a point, the editor uses the most recently entered signal name. This is useful if you place the first two signal names incorrectly. If the attachment lines show that the names are not connected to the appropriate signals, use UNDO to delete the incorrect attachments, then type BUSNAME and click on the drawing area. You can then reposition the first two names.

Related Commands

Signame
Bustap

Bustap

Syntax Procedure

Syntax

```
BUSTap bus_tap_value point point...
```

Description

This command fills in the value of the BN (bit number) properties on bus taps in drawings.

The bus_tap_value should be in the Design Entry HDL signal syntax with the first number as the start tap value, the second number as the end tap value, and the third number the increment. For example, a bus_tap_value of 7..0:2 results in BN values of <7>, <5>, <3>, and <1>, with <7> being placed on the tap closest to the first point, <1> on the tap closest to the second point, and <5> and <3> on the taps in-between.

The third number is optional. If no third number is specified, for example, 7..0, an increment of 1 is assumed. The second number is also optional, and if not specified, for example `7', all taps pointed to will have a BN value of the first number.

Bwire

Syntax Procedure Related Commands

Syntax

```
Bwire {[signal_name] point  point }  . . .
```

Description

The `bwire` command adds a net between point/component/block and a block. When the net is added, a pin also gets attached at the connection point in the block.

See also the WIRE, BROUTE, ROUTE commands, and the SET and SHOW commands to change default wiring behavior.

Example

To create a net `AB` and a pin `AB` on a block in the design:

1. Type `bwire AB` in the console window.
2. Start at some point and end on a block in the design.

   A net named `AB` and a pin on the selected block named `AB` are created.

Related Commands

Broute
Route
Set
Show
Wire

Change

Syntax Procedure

Syntax

```
Change {group_name | point}...
```

Description

This command modifies selected lines of text in place. The selected text items are highlighted, and the cursor is placed on the first text item. In case of notes, the cursor is placed before the character you click.

For properties, if both the name and the value is visible, the cursor is placed after the = character. If only the name or the value is visible, the cursor is placed at the beginning. After changing one line of text, the user can move over to the next text item by typing <cr>. The changes made to the line of text are then committed. The changes made to each line of text can be undone using the UNDO command.

Example

Type `change` and click a note in the design window. The cursor is placed before the character you clicked in the note. If you clicked the beginning of the note, the cursor is placed before the first character of the note.

The following commands are useful for editing text:

<BackSpace>	Deletes the previous character
<Delete>	Deletes the current character
Ctl<k>	Deletes to the end-of-line
Ctl<e>	Brings up the selected text items in an editor
<-	Moves backward
->	Moves forward

Check

Syntax Related Commands

Syntax

```
CHEck <cr>
```

Description

This command checks for connectivity problems and general errors on the current drawing. An option exists to allow the user to turn the check on or off. Design Entry HDL performs the following checks on the schematic:

- Duplicate components in the same location

  set option: CHECK_PARts_at_same_loc <ON/OFF>

  To correct the error, use the `split` console command and separate overlapping elements. For more information on the `split` console command, see Split on page 130.

- Pins attached to more than two wire segments (this may not be an error, but is an error if a wire inadvertently shorts the pins on a device) set option: CHECK_TWo_wires_at_pins <ON/OFF>

- Wires connected to only one pin and not named (NC wires)

  set option: CHECK_Unconn_wires <ON/OFF>

- Nets that are named but not connected to any pins

  set option: CHECK_SIGNAMES <ON/OFF>

- Wires that come close to but do not contact pins

  set option: CHECK_PIN_near_wire_endpt <ON/OFF>

- Missing TITLE and/or ABBREV properties

  set option: CHECK_TItle_abbrev <ON/OFF>

- Bodies that are placeholders

  set option: CHECK_Body_place_holders <ON/OFF>

- Pins located at the origin (0,0) in BODY drawings

  set option: CHECK_PINS_at_origin <ON/OFF>

- Multiple dots at the same location

  set option: CHECK_Arcs_at_same_loc <ON/OFF>

  To correct the error, use the `split` console command and separate overlapping elements. For more information on the `split` console command, see Split on page 130.

- Hard properties with the? value (placeholders)

  set option: CHECK_Prop_place_holders <ON/OFF>

- Wires connecting the pins of a two-pin body

  set option: CHECK_Shorted_pin <ON/OFF>

- Wire segments hidden by parts of a body

  set option: CHECK_Hidden_wires <ON/OFF>

  To correct the error, use the `split` command and separate overlapping elements.

- Pin properties which are no longer attached to pins.

  set option: CHECK_Missing_pins <ON/OFF>

  To correct the error, reattach properties to new pins, delete properties or replace part.

- Inconsistent section properties

  set option: CHECK_Pack_sec_type_props <ON/OFF>

  To correct the error, resection the part.
- Signame properties defined within a symbol (body).

  set option: CHECK_SIGNAME_in_body <ON/OFF>

  To correct the error, remove signame properties from the body files.
- Duplicate PATH properties
- Wires overlapping a body
- Check for legal HDL net names (hdl_direct on only)

  set option: CHECK_Net_names_hdl_ok <ON/OFF>
- Check for legal HDL port names (hdl_direct on only)

  set option: CHECK_Port_names_hdl_ok <ON/OFF>
- Check for legal HDL symbol names (hdl_direct on only)

  set option: CHECK_Symbol_names_hdl_ok <ON/OFF>
- Run checks automatically when writing drawings.

  set option: CHECK_On_write <ON/OFF>

CHECK lists each detected error. After you run the CHECK command, you can use the ERROR command to locate each error on the drawing.

Related Commands

Error

Circle

Syntax Procedure Related Commands

Syntax

```
Circle point point
```

Description

This command adds circles to a drawing. To place a circle on the drawing, enter the CIRCLE command and select a point as the center of the circle. To size the circle dynamically, drag the mouse and then click again to place the circle on the drawing.

Circles and arcs are rarely necessary on logic designs but are commonly used for creating symbol drawings.

Related Commands

Arc

Copy

Syntax Procedures

Syntax

```
COPy {[count][REPEAT][ALL] source_point destination_point | [count][REPEAT][ALL]
group_name destination_point | property_point destination_point attach_point }
```

Description

This command copies objects, properties, and groups in the current drawing or between viewports.

COUNT indicates the number of copies to place on the drawing. To make multiple copies, type COPY and enter a number to specify the number of copies to make. Move the cursor to the object or group to be copied and click to select an object or the center button to select a group. Click to place the copies at grid points or the right button to place copies at the vertex nearest the cursor. After you place the first copy, the remaining copies are automatically added to the drawing. The second copy is offset from the first copy by the same distance as the first copy is from the original. You can use this feature to copy single items and groups.

SOURCE_POINT is the object to copy, PROPERTY_POINT is the property to copy, and DESTINATION_POINT is the position point for the new copy. When you copy a property, ATTACH_POINT attaches the property to an object (symbol, pin, rewire).

To copy an object (such as a symbol or a wire), type COPY, position the cursor on the object, and press the appropriate button. The left button picks up a copy of the object at the grid point nearest the cursor. The right button picks up a copy of the object at the vertex nearest the cursor. (The vertex of the copy snaps to the cursor.) This is useful for copying component bodies and wires. Click to place the copy on the grid point nearest the cursor or the right button to attach the copy to the nearest vertex (useful for attaching copies of wires at new locations).

To copy a group, use the GROUP or SELECT command to define a group, then type COPY. Move the cursor to the group to be copied and click the center button to select the nearest group. You can also type the single-letter GROUP_NAME and press <cr>. Click to place the copy.

To copy properties, type COPY, and click to select the property to copy. Move the cursor to the location for the copy and click. A flexible line is drawn from the property to the cursor. Move the cursor to the object where the property is to be attached and click. You can attach the property to a part, wire, pin, or signal name.

You cannot copy default symbol properties, soft properties, PIN_NUMBER properties, or properties generated by the SECTION, PINSWAP, and BACKANNOTATE commands. User-added properties are included in copies of parts. Signal names are not copied. Wire properties are not included when you copy a wire. If a default symbol property on a symbol was changed, the copy of the symbol contains the changed value.

There are two options to the COPY command. COPY ALL will copy section properties, soft properties, pin properties, wire properties, and properties attached to other properties. This option is especially useful if you are copying a section of logic from one drawing to another. If you want to place a copy of something in several unrelated places, try using the COPY REPEAT option. This option causes the copy command to reselect the objects you originally selected after you have placed a given instance. The REPEAT and ALL options may be used together.

Groups of properties are not copied. When applicable, properties attached to objects are copied with the group.

Procedures

- To copy an object
- To copy an object and its properties
- To make multiple properties of an object

Dehighlight

Syntax Procedure Related Commands

Syntax

```
DEHighlight [ Net | PArt | PIn | Any ] pt
DEHighlight [ Net | PArt | PIn | Any ] object_name
```

Description

Net, Part and Pin are used to specify the object type you want to unhighlight. You may use Any if you want all selected objects to be unhighlighted. This is also the default argument if you do not specify any object type

.

pt	To pick a component to be unhighlighted, point to the object and press the left button. Select a group by pointing to the required group and pressing the middle button.
object_name	If you would prefer typing in the name of the object, you may do so by using the second version of this command.

For nets, the object name is the signal name of the net, for example, FOO.

For parts, the object name is the value of the PATH property attached to the component, for example, 7P.

For pins, the object name is the value of the PATH property attached to that component to which the pin belongs, followed by a period ("."), followed by the name of the pin, for example, 7P.A<SIZE-1..0>.

Wildcards such as * and ? may be used in specifying the object name,

For example, FOO*, 7*P, 7P.A*.

The Dehighlight command is used to unhighlight an already highlighted object throughout the system.

Related Commands

Highlight
Unhighlight

Delete

Syntax Procedures Related Commands

Syntax

```
DELete {point | group_name}...
```

Description

This command removes objects from a drawing. To delete an object, point to any part of the object and press the left button. To delete a group, use the center button or type in the single-letter group_name. DELETE removes the object or group nearest to the cursor.

You cannot delete default properties on bodies and pin number properties generated by the PINSWAP command.

The UNDO command lets you retrieve groups or objects deleted by mistake.

Example

To delete an object:

1. Type `delete` in the console window.
2. Point to an object in the design window and press the left button.

Procedures

- Deleting an object
- Deleting a group

Related Commands

Undo

Diagram

Syntax Procedure

Syntax

```
DIAgram [<library>]cell[.type][.version][.page]
```

Description

This command works like the *File – Save As* menu option in Design Entry HDL. You can use an existing drawing as a pattern for a new drawing or save a copy of a drawing by a different name before making changes to it.

- `<LIBRARY>` is the name of the library where the drawing resides. The library name must be enclosed in angle brackets. If no library is specified, the current library is the default.
- `CELL` is the new name of the drawing. The current drawing name is taken by default. For example, if the Design Entry HDL title bar displays `ATM.SCH.1.2`, the current drawing name is `ATM`.
- `TYPE` is the drawing type. The drawing type can be `SCH` (schematic), `SYM` (symbol). If no drawing type is specified, the current drawing type is used. For example, if the Design Entry HDL title bar displays `ATM.SCH.1.2`, the current drawing type is `SCH`.
- `VERSION` is the version number of the drawing type. For example, if the Design Entry HDL title bar displays `ATM.SCH.1.2`, the version number of the drawing type `SCH` is `1`. If no version number is specified, the default value 1 is used.
- `PAGE` is the page number for the drawing. For example, if the Design Entry HDL title bar displays `ATM.SCH.1.2`, the page number of the current drawing is 2. If no page number is specified, the default value 1 is used.

To rename a drawing, edit the drawing to be changed, type DIAGRAM and the new name of the drawing. Type WRITE to save a copy of the drawing by its new name.

For example, to use the drawing `SHIFTER.SCH.1.1` as a pattern for a new drawing named `NEWSHIFTER.SCH.1.1`, use the commands:

```
EDIT SHIFTER.SCH.1.1
DIAGRAM NEWSHIFTER
WRITE
```

The `NEWSHIFTER.SCH.1.1` drawing is saved to disk.

Directory

Syntax   Related Commands

Syntax

```
DIRectory {[<directory>][name][.[type][.[vers][.[page]]]] |<cr>}
```

Description

This command lists the names and contents of directories in the current directory list in the order that the directories are searched with the current working directory displayed first.

DIRECTORY <cr> accesses the DIRECTORY BROWSER form. The DIRECTORY BROWSER form shows the current library or directory. Any items listed in the left section of the form are from the current directory or library. To list the contents of a different directory or library, select the Current Dir/Lib field and type the directory or library name. The directory form can be turned off by typing in SET DIRFORM OFF.

<DIRECTORY> is the directory name whose contents you want to list. The name must be enclosed in angle brackets. If no directory is specified, the current directory is taken by default. NAME is the drawing to be listed. Unless you specify the drawing type, the version number, and the page number, the DIRECTORY command displays only the drawing name. You can also list drawings by type, versions, or pages.

You can use wildcards in all fields of the directory and drawing names. An asterisk matches any string. A question mark matches any single character.

Some command examples are
:

DIR	Lists all drawing names in the current directory
DIR <*>	Lists all active directories (but no drawing names)
DIR <time>*	Lists all drawing names (parts) in the TIME library

DIR <*>*	Lists all drawing names in all active directories and libraries
DIR ls*	Lists real file name and directory type for current directory.
DIR *.*	Lists all drawings.

Related Commands

IGnore
Library

Display

Syntax   Procedures

Syntax

```
DISPlay   { Name | Value | Both | Invisible | Default | scale_factor |
Center_justified | Left_justified | Right_justified | Heavy | Thin | Pattern
<pattern_number> | Filled | Open }{ point . . . | group_name . . . }
```

Description

This command changes the way objects or groups are displayed on a drawing. Any change made with the DISPLAY command remains in effect until another DISPLAY command is used to change it again.

To change the display of a single object, use the left mouse button. To change the display of a group, use the middle mouse button or type in the single-letter group name.

Groups can contain any type of object. Group names, options, and point entries can be included in any order and in any combination, except that the first argument MUST be a command option.

The command options are described below:

- NAME, VALUE, BOTH, and INVISIBLE determine the way properties are displayed on the drawing. Although a property consists of a name and value pair, usually only the value is displayed when a property is added to a drawing. These options allow you to display the name alone, the value alone, both, or neither.
- DEFAULT and SCALE_FACTOR determine the size of text displayed on the drawing.

- ❑ DEFAULT displays text on the drawing using the default text size specified in the *Text* tab of the *Design Entry HDL Options* dialog box. For example, if the default text size is `0.082` inches, the size of the selected text on the drawing will be set to 0.082 inches.
- ❑ You specify a scale_factor to enlarge or reduce the size of the text on the drawing. For example, if the default text size specified in the *Text* tab of the *Design Entry HDL Options* dialog box is `0.082` inches, the `DISPLAY 2` command will enlarge the size of the selected text to `0.164` inches.

Note: To display the size of a string, type SHOW SIZE and point to the string. Use SET SIZE to alter the default size of added text.

- A text string added to a drawing is defined by a vertex at the lower left corner of the string. To change the justification, use DISPLAY CENTER_JUSTIFIED, DISPLAY RIGHT_JUSTIFIED, or DISPLAY LEFT_JUSTIFIED.
- HEAVY, THIN, and PATTERN change the way an existing wire appears on a drawing. Heavy makes the wire thicker making it look like a bus. Thin returns a heavy wire to the default wire thickness. Pattern changes a wire to one of six patterned lines. Pattern 1 is a filled line (the default); patterns 2-6 are a variety of dotted and dashed lines. In a LOGIC drawing, the entire net changes. In a SYMBOL or DOC drawing, only the wire segment specified by the cursor changes.
- FILLED and OPEN change the display of dots already added to a design.
- Open dots scale when the ZOOM or SCALE command is used; filled dots do not. The SET DOTS_FILLED command adds dots to the drawing filled by default.

Procedures

- Displaying properties on objects
- Resizing text
- Displaying properties on objects in a group

Dot

Syntax Procedure

Syntax

```
Dot point...
```

Description

This command adds dots to drawings to indicate connection points. Dots are used in logic drawings to indicate that lines crossing one another are connected. By default, lines crossing are not connected unless dotted. Wires joining at a 'tee' are connected, even without a dot. Dots are used in symbol drawings to indicate pin connection points.

Dots can be filled or open. By default, all added dots are open. To change to filled dots, type SET DOTS_FILLED. To fill an open dot, type DISPLAY FILLED and point to the dot.

AUTO DOTS places a dot at all connection points in a logic drawing. AUTO UNDOT automatically removes all dots except those at the intersections of four wires.

AUTO UNDOT removes all existing dots in the drawing.

Example

To indicate a connection point on a wire, type `dot` in the console window and click on the wire where you want the connection point.

Echo

Syntax

```
Echo message
```

Description

This command displays messages in the console window. Use this command to show messages during the execution of a script for tracking its progress and debugging.

Example

```
echo 50% script completed
```

Edit

Syntax   Procedures   Related Commands

Syntax

```
Edit {[<directory>][drawing][.[type][.[version][.[page]]]]|<cr> | point}
```

Description

This command displays an existing drawing to be edited or creates a new drawing.

EDIT <cr> accesses the *View Open* form that allows you to open an existing drawing for editing. The *View Open* form can be turned off by typing SET EDITFORM OFF.

To edit a drawing directly from the command line, type EDIT and the drawing name. <DIRECTORY> is the directory where the editor is to search for the drawing. If not specified, each directory in the list is searched until a drawing by that name is found. The directory name must be enclosed in angle brackets. DRAWING is the name of the drawing to edit. If the specified drawing is found, it is displayed on the screen. If it is not found, the system creates a new drawing by that name in the current library when you write the drawing.

The default value for both version and page is 1. Page specifications for symbol drawings are ignored, but each symbol can have multiple versions. Other drawing types can also have multiple versions and pages.

You can edit a second drawing without writing the current drawing. EDIT saves the first drawing, along with any changes, in a temporary file before bringing in the new drawing. If you edit the first drawing again, EDIT displays the modified version from temporary storage. The SHOW HISTORY command lists all drawings that have been edited during the current session and states whether they have been modified.

The EDIT command also allows you to examine the drawings associated with symbols on the screen. By default, the SYMBOL.CSS file of a hierarchical symbol is edited when you select the symbol from the current drawing. For example, to edit the logic associated with a SUBTRACTOR symbol in the current drawing, type EDIT and point to the symbol with the left button. The current drawing is placed in temporary storage, and the drawing SUBTRACTOR.SYM.1.1 is displayed for editing.

Procedures

- Navigating the drawing hierarchy
- Opening a drawing
- Creating a design page
- Displaying pages in a multi-page drawing

Related Commands

Get
Return
Gotosheet

Error

Syntax

```
Error <cr>
```

Description

This command locates and displays each error detected by the CHECK command. It draws a blinking highlighted rectangle at the location of the error and displays a message describing the error.

Exclude

Syntax Procedure Related Commands

Syntax

```
EXClude [group_name|Mpoint|DEFault][option...][selection ...| group_name]
```

where option is:

```
BOdies | WIres | PRoperties | NEts | Connections
```

and selection is:

```
Lpoint | Ctrl+Rpoint | Mpoint
```

Description

This command removes items or groups from the current group.

- If the first argument is a single-letter group name, the group will become the current group. Alternatively, click the middle mouse button on a highlighted group to make it the current group. If a group is not specified, or the word default is provided, the most recently created group will remain the current group.
- To remove individual objects, click the left mouse button or press *Ctrl* and click the right mouse button.
- To remove previously-defined groups, click the middle mouse button on a highlighted group or enter the single-letter group name.
- Option flags allow the user to remove types of objects in a group. Options are applied to the objects in the initial current group.

  Specifying BODIES, WIRES, or PROPERTIES removes all occurrences of the specified type from the current group. NETS is the same as WIRES. Specifying CONNECTIONS removes all symbol pins (but not the symbol origins) from the current group.

Example

```
Exclude A ne
```

This command excludes all nets in group A from the group.

Related Commands

Find
Group
Include
Select

Exit

Syntax Procedure Related Commands

Syntax

```
EXIt <cr>
```

Description

This commands terminates an editing session. The editor displays a message if there are unwritten changes to any drawings in the current editing session and asks you if you really want to quit. You must answer Y or YES to exit. Any other response aborts the command.

The QUIT command is the same as the EXIT command.

Related Commands

Quit

Filenote

Syntax Procedure

Syntax

```
FILenote {file_name point| <cr>}
```

Description

This command includes a named text file in a drawing at the specified point. POINT is the position in the drawing to add the text. When the file is added, each line in the file is converted into a note that can be individually moved, copied, deleted, or changed. Empty lines in the file are ignored. To include a blank line in the note, type a space on the line in the file.

FILENOTE <cr> brings up a file browser. Select the file and click the position on the drawing where you want to place the note.

Find

Syntax Procedure Examples Related Commands

Syntax

```
FINd pattern
```

Description

This command searches the current drawing and places all objects that match a specified pattern into a group. A pattern can match symbol names, notes, property names, property values, or signal names. You can search for properties by specifying both name and value separated by an equal sign.

Wildcards are allowed in a pattern. An asterisk matches any number of characters, and a question mark matches any single character. FIND is not case-sensitive.

All items found with the command are placed in a list. You can step through the list items using the NEXT command. This command places a blinking highlighted rectangle around each item on the display so that it can be changed or deleted.

By using the `SET NEXTgroup` command before the FINd command, you can add the results of the FINd operation to the specified group. Commands to do this are:

```
SET NEXTgroup <groupname>
FINd pattern
```

Examples

- To find all `ls04` components on a drawing and add them in group A, run the following console commands:

  ```
  set nextgroup A
  find ls04
  ```

- To find all components with any reference designator assigned, run the following command:

  ```
  find *LOCATION=*
  ```

- To find all objects on a drawing that start with the letter `mem` and add them in group A, run the following console commands:

  ```
  set nextgroup A
  find mem*
  ```

- Suppose you have five instances of ls04 with property LOCATION = U1 on a page. To change the `LOCATION` property of all these instances to U8, do the following:

  a. Type the console command:

  ```
  find LOCATION = U1
  ```

b. Type the console command:

```
change A
```

c. Right-click on the canvas and select *Editor* from the pop-up menu.

The text editor opens.

d. Change the value of `LOCATION` property to `U8` for all the objects.

e. Choose *File – Save* and save the file.

Related Commands

Group
Exclude
Include
Select
Set

Get

Syntax Procedure Related Commands

Syntax

```
Get { [<directory>][drawing][.[type][.[version][.[page]]]] }
```

Description

This command replaces the current copy of a drawing with the version stored on the disk. The fresh copy of the drawing replaces any previously read (and perhaps modified) version in the editor. GET is useful while editing a drawing if you want to discard the current work and go back to an earlier version.

Note: All modifications to the current drawing are lost when you use the `Get` command.

Example

To discard changes to your current drawing:

1. Type `get` in the console window and press `Enter`.

A Design Entry-HDL message box appears asking you to confirm if you really want to get the drawing from the disk.

2. In the Design Entry-HDL message box, click *Yes*.

Related Commands

Edit

_globalModify

Syntax

Syntax

```
_globalModify
```

Description

Opens the Component Change tabbed page of the Global Modification window. Use this page to replace a component with a new component across a design.

_globalChange

Syntax

Syntax

```
_globalChange
```

Description

Opens the Property Change tabbed page of the Global Modification window. Use this page to change properties of components, pins, and nets across a design.

_globalDelete

Syntax

Syntax

```
_globalDelete
```

Description

Opens the Property Delete tabbed page of the Global Modification window. Use this page to delete properties of components, pins, and nets across a design.

_globalBatch

Syntax	Command File Example	Related Commands

Syntax

```
_globalbatch
```

Description

Access to the batch mode operation is provided through a Design Entry HDL console command called `_globalBatch`. The _globalBatch command is used only for flat designs. This command takes a single argument, that is, the name of a command file. Relative paths are resolved according to the location of the CPM file.

A command file can contain a single command or as many commands as you want. All commands contained from within a command file are dumped to a single log file. If multiple log files are desired, you must use multiple command files. Command files can handle comments.

Command File Example

```
;; Sample Global Change/Delete/Modify/Replace Command File
     ;; A Semicolon NOT FOUND inside double quotes designates a comment
     ;; This file must contain 1 master structure but the structure can
     ;; contain as many commands as desired.
     ;; White space is ignored as long as it is NOT within double quotes
     ;;
     ;; The following are case insensitive keywords and do not need to be quoted:

;; True, False, Design, Page, Module
     ;;
     ;; All property names, values, component names, library names, component
     ;; versions and page ranges must be quoted.
     ;;
     ;; The -SCOPE option supports keywords or a range of pages. Even though
     ;; the keywords do not need quotes the rage range does. The page range
     ;; accepts comma separated list of pages and page ranges designated by a '-'

;; Example:  "1,3,5,7-12"
     ;;
     ;; A special keyword string "<<PRESERVE>>" is allowed in the _globalchange

;; -ToProp fields. This indicates to retain the source property name or
     ;; source property value. <<PRESERVE>> cannot be used for both the
     ;; name and value in the same run, otherwise there would be nothing to change!


     (  ;; The parenthesis starts the definition of the master structure
     (  _globalDelete
     ( -Nets     true            ) ;; True / False
     ( -Pins     true            ) ;; True / False
     ( -Comps    true            ) ;; True / False
     ( -Scope    design          ) ;; Design / Page / Module / "1,2,5-7"
     ( -Save     true            ) ;; True / False
     ( -Wild     true            ) ;; True / False
     ( -Prop     "name" "value"  ) ;; Double-Quoted Strings

     )  ;; Each command must also have starting and ending Parenthesis
     ;; This parenthesis ends the _globalDelete Command

     (  _globalChange
     ( -Nets      false          ) ;; True / False
     ( -Pins      false          ) ;; True / False
     ( -Comps     false          ) ;; True / False
     ( -Scope     page           ) ;; Design / Page / Module / "1,2,5-7"
     ( -Save      true           ) ;; True / False
     ( -Wild      true           ) ;; True / False
     ( -FromProp  "name" "value" ) ;; Double-Quoted Strings
     ( -ToProp    "name" "value" ) ;; Double-Quoted Strings or "<<PRESERVE>>"
```

```
    )  ;; This ends the _globalChange Command

    (  _globalModify
    ( -Scope     page           ) ;; Design / Page / Module / "1,2,5-7"
    ( -Save      true           ) ;; True / False
    ( -HardProp  true           ) ;; True / False
    ( -FromLib   "lib"          ) ;; Double-Quoted String
    ( -FromCell  "cell"         ) ;; Double-Quoted String
    ( -FromVer   "ver"          ) ;; Double-Quoted String
    ( -FromProp  "name" "value" ) ;; Double-Quoted Strings
    ( -FromProp  "name" "value" ) ;; Double-Quoted Strings
    ( -FromProp  "name" "value" ) ;; Double-Quoted Strings
    ( -ToLib     "lib"          ) ;; Double-Quoted String
    ( -ToCell    "cell"         ) ;; Double-Quoted String
    ( -ToVer     "ver"          ) ;; Double-Quoted String
    ( -ToProp    "name" "value" ) ;; Double-Quoted Strings
    ( -ToProp    "name" "value" ) ;; Double-Quoted Strings
    ( -ToProp    "name" "value" ) ;; Double-Quoted Strings

    )  ;; This ends the _globalModify Command

    ( Exit )
    )  ;; This parenthesis ends the definition of the master structure
```

Gotosheet

Syntax   Procedure   Related Commands

Syntax

```
GOtosheet N
```

where `N` is the page number in a hierarchical design.

Description

This command allows you to go to a specific page in a hierarchical design.

This allows you to easily refer to a page in Design Entry HDL against a plotted page or a cross-reference report.

- When you plot a schematic page, the page number of the schematic page is plotted if you have added the custom text variables for page numbers on the schematic page.
- When you cross-reference nets on a design, the cross-reference reports display the schematic page numbers.

Enter the page number displayed on the plotted page or the cross-reference report to go to the page in Design Entry HDL.

Related Commands

Edit

Grid

Syntax   Procedure   Related Commands

Syntax

```
GRId {<cr>|[ON][OFf][Dots][Lines] [grid_size grid_multiple]}
```

Description

This command alters the grid display. A grid helps you place objects and ensure wire alignment and pin connections. The GRID command options can be used to turn the grid on or off, change the display to solid lines (the default) or dotted lines, and alter the default spacing. GRID <cr> toggles the grid on and off or you can specify ON or OFF. The current values of the grid spacing are displayed on the status line at the bottom of the screen.

`Grid_size` specifies the separation of the grid lines. `Grid_multiple` indicates how many lines of the grid are skipped before the next line is displayed. The default value for LOGIC drawings is 5 (2 for SYMBOL drawings). You can specify a positive integer to change the default grid multiple. Specify 1 to display every line, 2 to display every other line, and so on. Be aware that if you change the grid size, objects that were previously on a grid location may now be off-grid and wires may not be connected even if they appear so. This is why you should use the right mouse button to connect wires to pins and other vertices.

See also the `SET` command to change the default editor values.

Related Commands

Set

Group

Syntax Procedure Related Commands

Syntax

```
GROup [group_name|DEFault][type...] {selection...| group_name | ALL}
```

where type is:

```
BOdies | PRoperties | NOtes | WIres | DOts.
```

and selection is:

```
{Lpoint Lpoint...Ctrl+Rpoint} | Ctrl+Rpoint | Mpoint
```

Description

This command allows the user to draw a polygon to specify the boundaries of a group. The group is defined as a collection of objects.

- If the first argument is a single-letter group name, that group will become the current group. If a group name is not specified, or DEFault is specified, a single-letter group name will be automatically assigned to the group that is created.
- The group selection can be restricted to a specified type or set of types by providing one or more of the type arguments.

  For example, the command `group A bodies properties` includes only the components and properties among the objects you have selected for grouping in group A, even though there are notes, wires or dots among the objects you have selected for grouping.
- You can use the mouse to draw a polygon around the objects to be grouped. Click the left mouse button to start the line or to change the direction of the line. Complete the polygon by pressing *Ctrl*+right mouse button when the cursor is near the starting point. You can draw additional polygons to include other objects in the group.
- Press *Ctrl* and click the right mouse button to include individual objects in the group.

- Click the middle mouse button on another highlighted group to include its contents in the current group.
- Use the name of a previously created group to include its contents in the current group.

  For example, the command `group C A` includes the contents of group A in group C.
- Use ALL to select all objects in the current drawing.

  For example, the command `group A bodies all` includes all the components in the current drawing in group A.

The console window displays the group name and the number of bodies, properties, notes, dots, and wires in the group.

Related Commands

Auto
Find
Include
Exclude
Select

HIEr_write

Syntax

```
HIEr_write [-quit]
```

Description

This command writes the hierarchical blocks in a schematic starting from the level of hierarchy from which it is executed. For every block, the *hier_write* command writes all the page and netlisting files. It then generates marker files for every block in the schematic in the `temp/hierwrite` directory of the design.

The *hier_write* command is different from the write command as *write* command saves only the current page while *hier_write* saves all the pages in the hierarchy.

the *-quit* option when used with *hier_write* command saves all the pages and quits DEHDL.

Highlight

Syntax Procedure Related Commands

Syntax

```
HIGhlight [Net | PArt | PIn | Any] pt
HIGhlight [Net | PArt | PIn | Any] object_name
```

Net, Part and Pin are used to specify the object type you want to highlight. You may use Any if you want all selected objects to be highlighted. This is also the default argument if you do not specify any object type.

pt	To pick a component to be highlighted, point to the object and press the left mouse button. Select a group by pointing to the desired group and pressing the middle mouse button.
object_name	If you prefer typing in the name of the object, you may do so by using the second version of this command.

For nets, the object name is the signal name of the net, for example, FOO.

For parts, the object name is the value of the PATH property attached to the component, for example, 7P.

For pins, the object name is the value of the PATH property attached to the component to which the pin belongs, followed by a period ("."), followed by the name of the pin, for example, 7P.A<SIZE-1..0>.

Wild cards such as "*" and "?" may be used in specifying the object name, for example, FOO*, 7*P, 7P.A*.

Description

The Highlight command is used to select objects in one tool and for operations by other tools in the system. For example, you could type in the `open` command in the simulator and then use the `highlight net` command in Design Entry HDL, to open a particular signal. Highlight is also used to allow the user to co-relate the same nets, parts and pins in the system. Thus, you could have your PCB layout tool and Design Entry HDL up at the same time and use the highlight part command in Design Entry HDL to highlight the component in both Design Entry HDL and the PCB layout tool.

If an object is already highlighted and you pick the object in the highlight pt version of the command, the object gets unhighlighted in the system. Thus, if you have selected a whole set of nets to open in the simulator, you could just double-click on the nets you want to select in Design Entry HDL and the net will be selected, but will not be left highlighted. However, if you type in the name of the object, it will be highlighted, even if it was already highlighted.

Related Commands

Dehighlight
Unhighlight

HMirror

Syntax

```
HMirror point
```

Description

This command is used to create a mirrored version of a selected symbol about horizontal axis (x-axis). To create a mirrored version of a symbol, type hmirror in the console window and select the symbol with the left mouse button.

Related Commands

Mirror

HPlot

Syntax

```
hplot [worklib/<schematic_name>/sch_1/module_order.dat]
```

Description

This command plots the hierarchical blocks in a schematic starting from the level of hierarchy from which it is executed.

In hierarchy mode, *hplot* plots a hierarchical block only once even if the block has multiple occurrences in a design. In occurrence edit mode, *hplot* plots all the occurrences of a hierarchical block in a design.

If you run the *hplot* command without any argument, it prints all the blocks in the schematic by reading the `pc.db` file of the schematic. If you specify the path of the `module_order.dat` file while executing the *hplot* command, *hplot* plots the blocks in accordance with the order specified in the file.

IGnore

Syntax

```
IGnore {directory_name | library_name | * | <cr> }
```

Description

This command causes a specified library and all its symbols to be deleted from the active search list. IGNORE <cr> ignores the design library. IGNORE * ignores all the project libraries in the current search list.

The IGNORE command prompts you to confirm that a library is to be removed from the search list. Click *Yes* to ignore the library. If the current drawing contains a part from the ignored library, that part is deleted from the screen and either turned into a placeholder or replaced if there is a part by the same name in another active library.

Example

To delete the `lsttl` library from the active search list:

1. Type `ignore lsttl` in the console window.
2. In the Design Entry-HDL confirmation window, click *Yes*.

imginsert

Syntax

```
imginsert [{filename} (starting coordinates)]
```

Description

Inserts an image into the schematic canvas. This command launches the Open dialog box from where you can select the image that you want to paste.

You can also insert an image at a pre-defined location from the command line.

Example

```
imginsert c:/abc.bmp (1000, 1200)
```

This command places the bitmap file, `abc.bmp`, at the specified location on the schematic.

imgstretch

Syntax

```
imgstretch [(starting coordinates) (ending coordinates)]
```

Description

Stretches a selected image horizontally or vertically on the schematic. You can also specify the starting and ending coordinates for stretching the image.

Example

```
imgstretch (1000, 1200) (1500, 2500)
```

imgcapture

Syntax

```
imgcapture [(starting coordinates) (ending coordinates)]
```

Description

This command captures screen shots of a selected part on a schematic. When you capture an image, it is copied to the clipboard from where it can be pasted into any graphics editor or a graphics-aware text editor such as Microsoft Word.

Example

```
imgcapture (1000, 1200) (1500, 2500)
```

where (1000, 1200) represent the starting coordinates and (1500, 2500) represent the ending coordinates.

Include

Syntax Procedure Related Commands

Syntax

```
INClude [group_name|Mpoint|DEFault][option...][selection ...| group_name]
```

where option is:

```
BOdies | WIres | PRoperties | NEts | Connections
```

and selection is:

```
Ctrl+Rpoint | Mpoint
```

Description

This command adds items or groups in the current group.

- If the first argument is a single-letter group name, the group will become the current group. Alternatively, click the middle mouse button on a highlighted group to make it the current group. If a group is not specified, or the word default is provided, the most recently created group will remain the current group.
- To add individual objects, click the left mouse button or press *Ctrl* and click the right mouse button.
- To add previously-defined groups, click the middle mouse button on a highlighted group or enter the single-letter group name.
- Option flags allow the user to include types of objects in a group. Options are applied to the objects in the initial current group and to any additions made to the current group by the include command.
    - ❑ BODIES or WIRES include all bodies or wires that have properties already in the current group.

- ❑ PROPERTIES include all properties attached to objects already in the related group.
- ❑ NETS include all nets of wires attached to bodies or wires already in the related group.
- ❑ CONNECTIONS include all the objects connected to the pins of any symbol already in the chosen group.

Example

```
Include A pr
```

This command includes properties of all objects in group A to the group.

Related Commands

Exclude
Find
Group
Select

Library

Syntax Procedure Related Commands

Syntax

```
LIbrary {library_name | <cr> }
```

Description

If you have already added libraries to your project, this command refreshes and updates the cells in the library. If you have not added libraries to the project, this command adds the specified library to the search list.

LIBRARY <cr> accesses the *Search Stack* form that allows you to add or delete libraries in the active search list. The *Search Stack* form can be turned off by entering the command SET LIBFORM OFF.

To add a library directly, enter the LIBRARY command and the *`library_name`* value directly on the command line. The last library added to the list is searched after any previously specified directories are examined.

Note: To update a symbol from the disk, use the `symread` command.

Related Commands

Directory
IGnore
symread
Use

Loadstrokes

Syntax Procedure Related Commands

Syntax

```
LOADStrokes {strokes_file | <cr>}
```

Description

This command loads a user-defined *`strokes_file`*. Strokes are user-defined line drawings you create with the mouse. You can customize command entry in the editor by using strokes.

When you access the editor, a set of default strokes is initially read from the file <*`your_install_dir`*>/`tools/fet/concept/concept.strokes`. You can include the `LOADSTROKES` command in an input script file in the `START_CONCEPTHDL` section of the `.cpm` file. To specify the input script file, use the `input_script` directive.

To draw a stroke, press and hold the left mouse button and drag to form the required stroke. If the stroke you enter does not match an existing stroke, an error message is produced. If you produce a valid stroke, the related command is executed.

Strokes must be entered in the same direction as they were created. This allows you to have two strokes that look the same but that are bound to different commands. For example, you may have two different strokes that both appear as diagonal lines but are bound to different commands. The difference is that one stroke is drawn from upper left to lower right, and the other is drawn from lower left to upper right. When you use the editor to view strokes, a cross signifies the starting point of the stroke.

To create your own *`strokes_file`*, or edit an existing *`strokes_file`*, you use the Stroke Editor. This system utility is used to create strokes for many applications.

LOADSTROKES <cr> brings up the File Browser form. You can then select the *`strokes_file`* from the form.

Related Commands

Strokefile

Mirror

Syntax Related Commands

Syntax

```
MIrror point
```

Description

This command is used to create a mirrored version of a selected symbol. If editing a symbol drawing, this command will not mirror all lines and arcs in the drawing about the Y axis. Justified text is shifted from left to right or right to left in the mirrored version. No other rotation is done.

The MIrror command should be used with caution, especially with bodies with unmarked pins, such as merge bodies. Reversing the bits causes subtle, hard-to-find errors in the design.

See SET LEFT/RIGHT and DISPLAY LEFT/RIGHT for justifying text, and the ROTATE, SPIN, and VERSION commands.

Example

To create a mirrored version of a symbol, type `mirror` in the console window and select the symbol with the left mouse button.

Related Commands

Display
Rotate
Set
Spin
Version

Modify

Syntax Procedures

Syntax

```
MODIfy point
```

Description

This command modifies a selected part. Design Entry HDL displays the Modify Component dialog box, which lists all the parts that satisfy the selection criteria of the properties attached to the selected component.

If the old part and the new part have the same property names, the value of the property in the old part is replaced with the value of the property in the new part.

Error messages, if any, are displayed in the DE-HDL console window.

Example

1. To modify a part, type `modify` or `MODI` in the console window.
2. Click the part you want to modify in the design window.

   The Modify Component window appears.
3. Select the new part in the Modify Component window.
4. Click *Close*.

   The old part is replaced with the new part you selected in Step 3.

Procedures

- Modifying an object
- Modifying components in a group

Move

Syntax Procedures

Syntax

```
MOVe {source_point destination_point|group_name destination_point}
```

Description

This command moves objects from one position to another in the current drawing or between drawings in different viewports. MOVE operates on groups or individual objects. `Source_point` is the object to move. `Group_name` is the name of a group to move. `Destination_point` is the new position of the group or object. Properties (excluding `PATH` properties) attached to objects are moved with objects. Properties can also be moved independent of objects. To undo the effect of moving objects from one viewport to another viewport, an undo command must be entered in each viewport.

To move a single object, type MOVE and position the cursor on the object to move. Press the left button to pick up the object that is nearest to the cursor (regardless of the grid setting) or the right button to pick up an object's vertex nearest the cursor. A vertex is defined as a symbol origin, symbol pin, a wire end, or a note origin. The right button is useful for moving bodies or off-grid objects. Move the object to its new location and press the left button to place the object on the grid point nearest the cursor or the right button to attach the object to the nearest vertex. Note that when using the right button for the source_point, any visible object is selected. The right button as a destination_point only considers symbol pins, symbol origins or wire ends as attachment points.

To move an object from off-grid to on-grid, type *Move* in the console window and press Enter. Keeping the CTRL key pressed, right-click on an off-grid component. The component will attach to your mouse pointer. To place the symbol on grid, (left-) click on the canvas. The symbol is placed at the closest grid point.

To move a defined group, specify the name of the group or press the center button to select the group to move.

MOVE preserves electrical connectivity when there are electrical connections (wires) leading to moved objects or groups. You can automatically re-route a wired part you have moved to another area of your design. You can move the part into place with the wires connected directly or orthogonally, or without wires as follows:

The first click of the middle button (while dragging) changes the shape of the wire from orthogonal to direct.

The second click of the middle button detaches the part from any wires and allows you to move the part freely.

The third click of the middle button re-attaches the wires and lets you drag the object the usual way until you place the object with the left button.

The wires are automatically re-routed only when you place the part with the wires connected directly (non-orthogonally) and the SET option AUTOROUTE is on (SET AUTOROUTE ON).

To move a whole wire and any attached object around the screen, select the middle of the wire. To lengthen a wire, select the outer third of the wire. When a wire is attached between two objects, you can move the wire and one object independently of the other object by selecting the wire nearest the object you want to move.

Procedures

- Moving text, wires, or an unwired component
- Moving a wired component
- Moving multiple objects
- Moving a group

Netrename

Syntax

```
_netrename <old_net_name> <new_net_name>
```

Description

You can rename signals using the popup menu. Select the signal that you want to rename, right-click and use *Rename Signal*. When you rename a net, all its associated constraints and properties are retained.

You can also use the `_netrename` console command to rename a net using the following syntax: `_netrename <old_net_name> <new_net_name>`

When renaming a net, the net must be present in the design block in which it is being renamed. For example, when a local signal, `CLK`, is renamed in the `full_adder` block, the signal will only be renamed in `full_adder`. If the signal is in multiple pages, it will be renamed across all the pages.

Multiple-bit vector signals can be renamed only if the new vector signal has the same width.

After renaming nets, it is recommended that you perform an explicit Electrical Constraint Set (ECSet) audit in Constraint Manager.

Design Entry HDL does not support the following:

- Nets cannot be renamed when working with read-only pages or when the design is in use by another user in a team design environment.
- The signal scope cannot be modified when renaming a net. For example, a global signal cannot be renamed as a local signal, or vice versa. If, however, you rename a signal that involves a scope change, you will be informed that the net will be renamed but the scope of the old signal will be retained. You can then proceed with or cancel the net renaming operation.
- Only entire buses with the same width can be renamed. Individual bus bits cannot be renamed. For example, you can rename Z10<3..0> to A12<3..0>, or Z10<3..0> to A12<0..3> but Z10<0..3> cannot be renamed to A12<4..8>.
- You cannot rename scalar signals to single-bit vector signals and vice versa.

Important

> Net renaming operations are auto-saved. Any unsaved changes prior to the net rename auto-save are saved at the same time. You cannot undo a net renaming operation.

For related information, see `ASK_RENAME_SIGNAME_OPTION` in *Allegro Front-End CPM Directive Reference Guide*.

Next

Syntax Related Commands

Syntax

```
NExt  <cr>
```

Description

This command displays the items located by the FIND command. The NEXT command traverses the list of items found by the FIND command and draws a blinking highlighted rectangle around the item. You can perform an operation on the object and then issue the NEXT command to proceed to the next item. You can step through the list only once.

NEXT cannot be used after the CHECK command. Use ERROR after the CHECK command.

Related Commands

Check
Error
Find

Note

Syntax Procedure Related Commands

Syntax

```
Note text_line... point...
```

Description

This command adds text strings to a drawing. Notes are text strings that appear on the drawing but do not affect the evaluation of the drawing. They are used to document a drawing. There are two ways to add notes to a drawing:

- Specify the points on the drawing where the notes are to be located and then type in the text. Press <cr> after each note to position each note on the drawing. As long as there are points remaining, the editor interprets entered text as notes to the drawing.
- Type in each line of text first and press <cr>. (You can enter several strings before placing them.) Then use the cursor and the left button to indicate where each note is to appear on the drawing.

Place quotes around notes beginning with an opening parenthesis. Notes within quotes are not interpreted as commands.

Related Commands

Filenote

Page commands

You can renumber the pages of a design through the _Page and Page commands:

_PAGE commands

- _PAGEInsert
- _PAGEDelete
- _PAGECompress
- _PAGEMove

_PAGEInsert

Inserts pages before or between existing or non-existing pages in a schematic. All subsequent pages are renumbered automatically and you need not worry about renumbering them manually. The maximum number of pages that you can insert in a single command is 250.

```
_PAGEInsert <Number_of_pages> <Location> [-nosave] [-noconfirm]
```

Number_of_pages	Indicates the number of pages to insert. The valid range is 1 - 250. This parameter must always precede the Location parameter.

Location

Indicates the location from where you want to insert the page(s). Pages are always inserted before the current page at the target location.

To add pages at the end of the schematic, you would specify *Location* as one greater than the last page number. If you specify *Location* as more than one greater than the last page number, a page gap is created between the current last page and the first of the newly inserted pages.

See Inserting Pages at the End of a Schematic.

-nosave

Keeps the newly inserted pages from being saved. Use this parameter to create page gaps in a schematic. This parameter can be placed anywhere in the argument list.

-noconfirm

Bypasses the message prompting you to confirm before executing the command. If you choose *Cancel*, the command is terminated. This parameter can be placed anywhere in the argument list.

Note: : Existing gaps in page numbers do not affect the *_PAGEInsert* command. If you insert a new page at a location where there is already a page gap, the command adds an additional page and the size of the page gap is unchanged.

Examples

The following examples of the *_PAGEInsert* command are explained in the context of different scenarios:

- Inserting Pages between Two Pages
- Inserting Pages at the End of a Schematic
- Inserting a Page Gap between Two Pages
- Inserting Pages Beyond the End of the Schematic

Inserting Pages between Two Pages

Initial page sequence: 1-10

```
_PAGEInsert 5 4
```

This command will insert 5 pages before page 4. The current page 4 will become page 9. All the pages will be saved and the corresponding Page* files will be created under the sch_1 directory. Finally, a message will display a summary of the newly inserted pages.

New page sequence: 1-15

Inserting Pages at the End of a Schematic

Initial page sequence: 1-15

```
_PAGEInsert 5 16
```

This command will add 5 pages starting from page 16. All the pages will be saved and the corresponding Page* files will be created under the sch_1 directory.

New page sequence: 1-20

Inserting a Page Gap between Two Pages

Initial page sequence: 1-3, 6-10, 12-15

```
_PAGEInsert 3 8 -nosave
```

This command will:

- Move pages, page 8 onwards, by three places to accommodate the new pages. As a result, the existing page 8 will become page 11.
- Insert a three-page gap - 8,9,10.

  If you click the *Next Page* or *Previous Page* buttons to move to other pages, you will be prompted to save page 8. If you choose to save this page, the corresponding `Page8.*` files will be created and the page number count will increase by one.

  New page sequence: 1-3, 6-8, 11-13, 15-18

  However, if you choose not to save page 8, the result of the *_PAGEInsert* command would be a 3 page gap in the schematic at the point of insert.

  New page sequence: 1-3, 6-7, 11-13, 15-18

Inserting Pages Beyond the End of the Schematic

Initial page sequence: 1-3, 6-10, 12-15

The *-nosave* option is of no value for this case as everything beyond the end of the schematic module is already blank.

```
_PAGEInsert 2 25 -noconfirm
```

This command will insert 2 pages starting from page 25. There will be a 9 page gap between pages 15 and 25. The confirmation message will be suppressed and the pages will be inserted without prompting you for confirmation

New page sequence: 1-3, 6-10, 12-15, 25-26

_PAGEDelete

You delete existent or non existent pages from a schematic by using the *_PAGEDelete* command. Unlike the existing *PAGE DELete* command, which leaves a page gap when you delete a page, the *_PAGEDelete* command does not create any page gaps, by default. As a result, the subsequent pages are automatically adjusted to reflect the new page sequence.

Syntax

```
_PAGEDelete <List_of_Pages> [-retain] [-noconfirm]
```

List_of_Pages	Indicates the pages to delete. You can specify an explicit number or a range of numbers. For example, 1,2,3,5-7, is a valid range. Spaces are not allowed between page numbers.
-retain	Retains the physical page numbers of the pages affected by the deletion. This parameter can be placed anywhere in the argument list.
-noconfirm	Bypasses the message prompting you to confirm before executing the command. If you choose *Cancel*, the command is terminated. This parameter can be placed anywhere in the argument list.

Note: The *_PAGEDelete* command can be used to remove specific page gaps. To remove all page gaps, use the *_PAGECompress* command.

Examples

The following examples of the *_PAGEDelete* command are explained in the context of different scenarios.

- Deleting Existing Pages

- Deleting Non-existent Pages
- Deleting Non-existent Pages out of Page Range
- Deleting Pages to Retain Physical Page Numbers of the Pages Following the Page(s) Being Deleted

Deleting Existing Pages

Initial page sequence: 1-15

```
_PAGEDelete 6-8
```

This command will delete pages 6, 7, and 8. The physical page numbers of all the pages following page 8 will be moved in by 3.

New page sequence: 1-12

Deleting Non-existent Pages

Initial page sequence: 1-3, 6-10, 12-15

```
_PAGEDelete 4 -noconfirm
```

If you try to delete a non existent page, the *_PAGEDelete* command will reduce the page gap between pages 3 and 6 by 1. Consequently, the pages after page 4 will move in by 1. The confirmation message will be suppressed and the pages will be deleted without prompting you for confirmation.

New page sequence: 1-3, 5-9, 11-14

Deleting Non-existent Pages out of Page Range

Initial page sequence: 1-3, 6-10, 12-15

```
_PAGEDelete 48
```

If you try to delete a non existent page, which is out of the page range of the schematic, it will result in an error:

Result: Command cannot be executed

Deleting Pages to Retain Physical Page Numbers of the Pages Following the Page(s) Being Deleted

Initial page sequence: 1-15

```
_PAGEDelete 6-8 -retain
```

This command will delete pages 6, 7, and 8. However, the physical page numbers of all the pages following page 8 will be retained. A gap of three pages will be created between pages 5 and 9.

New page sequence: 1-5, 9-15

_PAGECompress

You remove all the page gaps in a schematic by using the *_PAGECompress* command.

Syntax

```
_PAGECompress
```

Note: The *_PAGECompress* command does not accept any arguments.

Example

Initial page sequence: 1 4-5 10-13 15-17

```
_PAGECompress
```

New page sequence: 1-10

_PAGEMove

You move a sequence of pages to existent or non-existent locations in a schematic by using the *_PAGEMove* command. Unlike the existing *PAGEMove* command, which moves pages only at non existent locations, the *_PAGEMove* command allows page movement to existing pages of a schematic. Also, unlike the existing *PAGEMove* command, which allows only a single page move at a time, the *_PAGEMove* command allows you to move multiple pages, simultaneously. In addition, you can move non-contiguous pages to contiguous locations.

The *_PAGEMove* command works as a drag-and-drop functionality in a GUI, and does not create any page gap for the moved pages. As a result, the total page count remains the same. However, gaps existing in page numbers before the move command is executed are retained.

Syntax

```
_PAGEMove <List_of_Pages> <Before_Page > [-noconfirm]
```

List_of_Pages	Indicates the pages to move. This parameter must come before the *Before_Page* parameter.
Before_Page	Indicates a location in the schematic where the pages will be moved. For example, if you specify 5 as *Before_Page*, the moved pages will precede the current page 5. To move pages to the end, specify a number, which is one greater than the last physical page.
-noconfirm	Bypasses the message prompting you to confirm before executing the command. If you choose *Cancel*, the command is terminated. This parameter can be placed anywhere in the argument list.

Examples

The following examples of the *_PAGEMove* command are explained in the context of different scenarios:

- Moving a Page before an Existing Page
- Moving a Set of Pages outside the Current Range of Pages
- Moving Non-contiguous Pages to Contiguous Locations
- Moving a Set of Pages to a Location which Falls within the Range of the Pages to be Moved

Moving a Page before an Existing Page

Consider a team design scenario where you have a team of three designers A, B, and C, working on a design. Each designer owns different sections of the design:

- Designer A is writing the CPU logic in pages 1-10
- Designer B is writing the CONTROL logic in pages 11-20
- Designer C is writing the MEMORY logic in pages 21-30

After integrating the design, as a Team Lead, you realize that a part of the CPU logic created in pages 4-8 should be moved after the CONTROL logic between pages 14 and 15. You decide to move pages 4-8 by executing the following command:

```
_PAGEMove 4-8 15
```

This command:

1. Moves pages 15-30 out 5 pages to make space for the 5 pages being moved.
2. Moves pages 4-8 to the blank slots created in step 1
3. Removes the 5 page gap (4-8) created in step 2 by moving all pages in by 5.

Moving a Set of Pages outside the Current Range of Pages

Initial page sequence: 1-3, 6-10, 12-15

```
_PAGEMove 6-9 17
```

This command will move pages 6-9 before the non-existent page 17 inserting pages backwards from page 16. Therefore, initial page 9 will be moved to page 16, the initial page 8 will be moved to page 15, and so on and so forth. All existing page gaps are maintained, but the page gaps created in the process of the move command are closed.

New page sequence: 1-3, 6, 8-11, 13-16

Moving Non-contiguous Pages to Contiguous Locations

Initial page sequence: 1-3, 6-10, 12-15

```
_PAGEMove 3,7,9 15 -noconfirm
```

This command will move pages 3, 7, and 9 before page 15. Pages 3, 7, and 9 will move to Pages 12, 13, and 14, respectively, and the other pages will be adjusted accordingly. The confirmation message will be suppressed and the pages will be moved without prompting you for confirmation.

New page sequence: 1-2, 5-7, 9-15

Moving a Set of Pages to a Location which Falls within the Range of the Pages to be Moved

Initial page sequence: 1-3, 6-10, 12-15

```
_PAGEMove 6-9 7
```

This command will result in an error as the target move location is within the range of the pages to be moved. A warning message is displayed and no action is taken.

Page

- page move
- page swap
- page delete
- page reset
- page forcereset

page move

```
page move <@lib.cell(view)> X Y
```

Moves existing page X to a non-existing page Y.

Note: You cannot move a page to a page that already exists.

page swap

```
page swap <@lib.cell(view)> X Y
```

Swaps existing pages X and Y.

page delete

```
page delete <@lib.cell(view)> X
```

Deletes an existing page X.

Note: The move, swap, and delete commands cannot be executed on pages if:

- the page is currently being edited.
- the page is being edited, and changes in the page have not been saved.

Note: You need not save or repackage the design after you run the move or swap command. This is because the canonical names in the design do not change after you run the move and swap commands. If you run the delete command, you must save and repackage the design.

page reset

```
page reset X
```

Sets the existing page number to X.

This command sets the logical page number of the currently open page to X. X is a logical page number. When this command is run, Design Entry HDL first checks the existence of the logical page number X in the physical page files. This command is executed only if the logical page X does not exist in any of the physical page files.

page forcereset

```
page forcereset all
```

with release 16. 5, the `page forcereset X` command is disabled because logical page duplication is not allowed. Now, `page forcereset all` is the valid workaround to adjust logical page number through the module.

You should keep the following points in mind while executing the `page reset` and `page forcereset` commands:

1. To commit the change made by the `page forcereset` or `page reset` command, you have to save the page.To reflect this status, a * is shown on the Design Entry HDL title bar. If you want to de-commit the change made by any of these commands, do not save the page because you cannot undo the command.
2. If two different physical pages have the same logical page number, and you try to save the design, Design Entry HDL displays an error message stating that the physical pages have the same logical page number. The message prompts you to use the `page forcereset` command and then save the schematic to resolve the page conflict.

   You can then use the `page forcereset` command to change the logical page numbers.
3. Running the `page forcereset` and `page reset` commands makes the occurrence properties of objects, which are on renumbered pages, unusable. A message box appears informing you that the `page reset` command might make the opf properties for objects on the page unusable. You get the following error message:
4. While cross-probing or globally locating objects, canonical names are shown with the logical page numbers. For example, the canonical name `@top_lib.top.(sch_1):page2_i5` displayed in the *Global Find* dialog box means that the component that has the `PATH=i5` property is located in the logical page 2 in the `sch_1` view of the cell `top` in the library named `top_lib`.

Paint

Syntax Procedures

Syntax

```
PAInt {color_name{point|group_name}|DEFault{point|group_name}} <cr>
```

Description

This command assigns selected colors to specified groups or objects. You can also enter the color_name on the PAINT command line, point to an object and press the left button. Select a group by entering the group_name or by pointing to the required group and pressing the center button.

PAINT DEFAULT paints objects or specified groups in their preset default colors. Use the SET COLOR commands (interactively or in the startup file) to establish default colors for the objects in your drawings. On a monochrome display, use the SHOW COLOR command to see what color an object is currently painted.

There are 16 available colors

- AQUA
- GREEN
- PINK
- SKYBLUE
- BLUE
- MONO
- PURPLE
- VIOLET
- BROWN
- ORANGE
- RED
- WHITE

- GRAY
- PEACH
- SALMON
- YELLOW

Procedures

- Specifying color of an object
- Specifying color for a group use

Pastespecial

Syntax

```
pastespecial [(starting coordinates) (ending coordinates)]
```

Description

Displays the *Paste Special* dialog box. Use this dialog box to specify whether you want to:

- Paste copied schematic parts on to the target schematic directly
- Change the signal names of the schematic before pasting them on the target location

Pause

Syntax

```
PAUse
```

Description

This command temporarily interrupts the editor until you press a key. PAUSE is useful in demos and scripts.

Pinnames

Syntax Procedure

Syntax

```
PINNames point
```

Description

This command adds a PIN NAMES symbol to a schematic drawing that defines the functional circuitry of a symbol drawing. This is used in hierarchical design and in library development.

The PIN NAMES symbol is added to an unused area of the schematic drawing. Design Entry HDL automatically attaches the names of the pins on the corresponding symbol drawing to the PIN NAMES symbol in the schematic drawing, and appends a `\I` suffix (scope = interface) to each signal name. The signal names can then be reattached to signals in the schematic drawing. The use of the PIN NAMES symbol eliminates the need to retype the signal names and reduces the chances of mislabeling signal names or omitting the interface scope (`\I`) signal property.

Pinswap

Syntax Procedure Related Commands

Syntax

```
PINSwap {point1 point2 | pin_number point }
```

Description

This command swaps the pin number defined to be in the same pin group. This command can only be used after initial pin number assignment using the SECTION command. Also, pin swapping can only occur between pins that have been defined in the library as swappable. For example, it may be legal to swap the two input pins of a NAND gate, but not the input and output pins of the gate.

There are two ways to swap pins:

- Type PINSWAP and point to the two pins to be swapped.
- Type PINSWAP, type in a new pin number, and then point to an existing pin. The selected pin is swapped with the pin having the pin number you specified.

The properties attached by the PINSWAP command cannot be changed, they can only be deleted and moved. Once pins on a part have been swapped, the part cannot be resectioned using the SECTION command.

The PINSWAP command also swaps sections within HAS_FIXED_SIZE parts.

Related Commands

Backannotate
Section

Plot

Syntax Procedure

Syntax

```
Plot
```

Plots the currently opened drawing

```
Plot cache
```

Plots all pages of the schematic named cache.

```
Plot cache.sym.1.1
```

Plots the symbol view of cache

```
Plot cache.sym.1.2
```

Plots page 2 schematic of cache

```
Plot cache.sch.1.*
```

Plots all pages of version 1

Note: To plot custom variables, you must open the design before running plot commands. Plot commands cannot evaluate the custom variables for unopened designs.

PPTAdd

Syntax Related Commands

Syntax

```
PPTAdd {"path"[",path"];}
```

Description

When using component selection in the physical mode, PPTADD provides the search path to locate the ppt files to be used.

The list of paths are pushed on to a stack. The last one specified is the first one searched.

This command replaces the SET PPTPATH option.

Related Commands

Add
PPTDelete

PPTDelete

Syntax Related Commands

Syntax

```
PPTDelete {"path"[",path"];}
```

Description

The PPTDelete command is used to remove paths to part table files used in the physical component selection. The command, however, does not affect the stack order.

Related Commands

Add
PPTAdd

PPTEcho

Syntax Related Commands

Syntax

```
PPTEcho
```

Description

When using the component selection in physical mode, PPTECHO lists the search path which will be used to locate the ppt files.

Related Commands

Add
PPTAdd
PPTDelete

Property

Syntax Procedure Dialog Box

Syntax

```
Property {attach_point location_point name_and_value |  name_and_value
attach_point  location_point  | attach_point  name_and_value  location_point}
```

Description

This command attaches a property name and a value to a specified vertex of an object. Properties allow you to associate information with selected objects on a drawing. The information is passed to other design programs for processing and analysis. A property consists of a name-value pair that is attached to an object (a symbol, pin, wire, or signal name). Operations on groups are not performed using the property command. Instead, use AUTO PROPERTY.

A property name can be any string of alphanumeric characters and underscores, provided that the first character is an alphabetic character. A property name cannot contain any spaces or punctuation marks except the underscore.

A property value can be any string of text up to 255 characters, including spaces and punctuation marks.

There are two ways to assign properties:

- Type PROPERTY <cr>

  Select the objects where properties are to be attached. Use the left button for objects. Select as many objects as the number of properties you want to attach. Type the name and value of the property, separated by a space or an equal sign. Press <cr> after each property entry. The properties are attached to the selected objects in the same order as the initial selection.

- Type PROPERTY <cr>

  Type the name and value of the property, separated by a space or an equal sign. Press <cr> after each property entry. Then specify the location on the drawing where the text of the property value should appear. Select as many objects as the number of properties you entered. The properties are attached to the objects you select in the same order as the initial property entries.

Each property attached to a given object, except the SIG_NAME property must have a unique name. If a newly entered property has the same name as a property currently attached to that object, the new property value replaces the old property value.

When a property is added to a drawing, only the property value appears. The SHOW PROPERTIES command temporarily displays the names and values of all properties on a drawing. The DISPLAY command changes the permanent display of property name and value pairs. The SET PROP_DISPLAY command controls the default display of added properties.

Quit

Syntax Related Commands

Syntax

```
QUIt  <cr>
```

Description

This command terminates an editing session. The editor displays a message if there are unwritten changes to any drawings in the current editing session and asks you if you really want to quit. You must answer Y or YES to quit. Any other response aborts the command.

The EXIT command is the same as the QUIT command.

Related Commands

Exit

Reattach

Syntax Procedure

Syntax

```
REAttach text_point  attach_point
```

Description

This command reattaches properties (including signal names) from one object to another. For example, you can use the REATTACH command to attach a property from the input pin to the output pin of a device.

To reattach a property, type REATTACH and select the property. A line is drawn from the property to the current cursor position. Specify the new attach_point for the property. Use the MOVE command to position the property at its new attachment point.

Recover

Syntax Procedure Related Commands

Syntax

```
RECover <recover_log_file>
```

Description

This command is used for recovering drawings that were being edited when Design Entry HDL or your system crashed.

Every time you start Design Entry HDL, a temporary directory is created in the `<project_directory>/temp` directory. By default, `xxnedtmp` is the name of the temporary directory. If the `xxnedtmp` directory already exists, a `xxnedtmp1` directory is created. If these two directories already exist, `xxnedtmp2` is created, and so on. An undo log file for each drawing is stored in this directory. The name of the undo log file for the first drawing edited is `undo1.log`. The second drawing's undo log file is `undo2.log`, and so on.

Note: The temporary directory (for example, ./`xxnedtmp`) and all of its files are deleted if Design Entry HDL terminates normally.

An example illustrating the use of the `recover` command is:

```
RECOVER temp/xxnedtmp1/undo2.log
```

The recovered drawing is given a unique name (for example `RECOVER1.SCH.1.1`) and is only saved in the memory (not on disk). You should use the `diagram` command to change the drawing name and use `write` command to write the drawing out to the disk.

Related Commands

Diagram
Write

Redo

Syntax Procedure

Syntax

```
REDo  <cr>
```

Description

This command reverses the last UNDO command. The system keeps a list of operations performed during the current editing session in a log file. The UNDO and REDO commands perform their functions according to this log file.

Remove

Syntax

```
REMove [<directory>][name][.[type][.[version][.[page]]]]
```

Description

Deletes a drawing from a design directory. REMOVE allows only one argument at a time. Repeat the procedure to delete additional drawings. If no directory is given, REMOVE searches for the specified drawing in the currently active design directory.

To delete a drawing, type REMOVE and the name of the drawing to be deleted, and press <cr>. The editor displays the names of the files to be deleted and asks you if you really want to remove the files. You must answer Y or YES to remove them. Any other response aborts the command. The directory entries are deleted, and the files are purged.

Wildcards are allowed in drawing_name. A question mark matches any single character, and an asterisk matches any number of characters. If only the drawing_name is specified, REMOVE deletes all drawing types (SYMBOL, LOGIC, SIM, and so on), versions, pages, and files (ASCII, binary, dependency, and connectivity) of the specified drawing in the directory.

Replace

Syntax Procedures

Syntax

```
REPlace {symbol_name point | symbol_name <cr> group_name}
```

Description

This command substitutes one part for another. The default version of the symbol is 1. Specify the version number to replace another version of a symbol.

There are several ways to use the REPLACE command.

- Type the name of the replacement part. Then use the cursor to point to the symbol or bodies to be replaced.
- Use the FIND command to group all the occurrences of a symbol to be replaced. Then, use the group_name option with the REPLACE command to globally change all the occurrences of the symbol. A message displays the number of bodies that are replaced.

Pin properties are reattached if a pin name on the new part is the same as a pin name on the first part. If the pin names do not match, the pin property becomes a symbol property.

All properties are retained except those generated by the BACKANNOTATE, SECTION, and PINSWAP commands. Unnamed signal names attached to the symbol are deleted. All default properties that have a value of ? receive the value of the property with the same name on the replaced symbol (if one exists). Wire connections to the original part are retained only if the pins are in the same location. The rotation of the original symbol is preserved when the symbol is replaced.

Procedures

- Replacing a component
- Replacing components of a group

Return

Syntax Procedure

Syntax

```
RETurn <cr>
```

Description

This command returns to the previously edited drawing. If the current drawing is modified but not written, the system saves a copy of the drawing before returning to the previous drawing.

The SHOW HISTORY command lists the drawings that you edited during the current session.

The SHOW RETURN command lists the drawings that the RETURN command will return to in the order that they will be accessed.

Rotate

Syntax Procedure Related Commands

Syntax

```
ROTate point
```

Description

This command rotates a symbol or text string by 90 degrees, with mirrors at 180 and 270 degrees. When a symbol is rotated, all notes and properties are also rotated and translated. You can then act on the properties independently.

To rotate a symbol or text string, type ROTATE and then point to the object to rotate. Each time you press the button, the part rotates 90 degrees. In the 90 -degree rotation, symbol notes are rotated 90 degrees to the left in their original justification.

Rotating some parts 180 degrees reverses the order of the pins. This can cause subtle errors in your designs if pins become incorrectly wired. To avoid this, a 180-degree rotation of a part becomes a mirror of a 0-degree rotation (about the Y axis). A 270-degree rotation of a part is a mirror of a 90-degree rotation (about the X axis). To get the other two rotations and the other two mirrors, use the MIRROR command to create another version of the device.

See SET and DISPLAY for justifying text.

Related Commands

Add
Display
Mirror
Set
Spin

Route

Syntax Procedure Related Commands

Syntax

```
ROUTE point point
```

Description

This command draws a wire connecting two selected points. The ROUTE command connects two points by drawing a series of orthogonal line segments between them. If it cannot determine a route, it draws a diagonal line directly between the two points. ROUTE will not run a wire through any existing objects or vertices.

To select the nearest pin or wire vertex for a ROUTE point, use the right button to select the point. Use any other button to select the nearest grid point.

Related Commands

Broute
Bwire
Wire

s2l

Syntax

```
s2l design
```

Description

This command runs the s2l (short2long) command on the current design. To update only the current page, type

```
s2l
```

The s2l command should be run on designs that have been upreved from SCALD to Design Entry HDL having property names that exceed 16 characters.

Before you run this command, you should execute the following two commands:

```
uprev design
uprev_write
```

In SCALD designs, property names in Design Entry had a limit of 16 characters. For a property name that has more than 16 characters, Design Entry HDL assigns a new property name with a shortened version and the original property name (over 16 characters) as its value. The original value, which has more than 16 characters, is entered by the user in <*your_install_dir*>/`tools/fet/pxl/allegroprp.dat`.

When Packager-XL is run on the SCALD design, it replaces the shortened property name with the original property name after finding it in `allegroprp.dat`.

For example, the property ELECTRICAL_CONSTRAINT_SET is shortened to ELECTRICAL_CONST by Design Entry HDL. Design Entry HDL also assigns ELECTRICAL_CONSTRAINT_SET as the value for the ELECTRICAL_CONST property. When Packager-XL is run on the design, it converts ELECTRICAL_CONST to ELECTRICAL_CONSTRAINT_SET and passes the design to Allegro.

In HDL, Packager-XL does not perform this conversion. Instead, this functionality of Packager-XL is handled by the s2l command entered in the Design Entry HDL console command window.

The character limit for a property name in Design Entry HDL is now 31.

Scale

Syntax Related Commands

Syntax

```
SCAle {point1 point2 drawing_name | drawing_name point1 point2}
```

Description

This command smashes a drawing and includes it in the current drawing. Point1 and point2 indicate the size of the rectangle where the smashed drawing will be placed. Drawing_name is the name of the drawing to smash. All bodies are turned into wires, arcs, and text. SCALE is useful for creating documentation drawings and new bodies.

When a drawing is smashed, all connectivity information is lost. The drawing can no longer be interpreted by the Compiler.

Related Commands

Smash

Script

Syntax Procedure Related Commands

Syntax

```
SCRipt { file_name | <cr> }
```

Description

This command performs the commands listed in the specified text file. Script files let you change the default editor behavior. SCRIPT allows you to operate in the batch mode using the same syntax as if you typed in the command. You can use the mouse to enter points, or you can specify the X-Y coordinates in the script file.

The syntax for specifying the points is:

```
(X Y)
```

where X and Y can range from +16000 to -16000.

For example, the command `vpdelete (0,0)` will delete the viewport that has the coordinates `(0,0)`.

You can configure a script to accept input during execution by including user input tokens in a script. User input tokens must be placed at the beginning of a new line. When the editor sees a user input token in a script, it highlights a menu button with the name of the editor command being executed. There are two user input tokens:

- $< When the editor encounters this token in a script, it prints from the token to the end of the text line as a prompt in the message window, and then waits for one item of input. The input can be a typed line, a function key press, a mouse action, or a *Ctrl+C*. You cannot use a <cr> as a response to a user input request.

- $; This token also prints from the token to the end of the text line as a prompt and awaits input. This token accepts and interprets inputs until you enter a semicolon. If this token is included, the editor follows the prompt with the message "Type ; when done with user input.”

To abort a script, press Ctrl+C. To abort at a user input token prompt, type a semicolon.

SCRIPT <cr> brings up the File Browser. The user can then select the names of the script file from the form.

The SCRIPT command will not recognize GED commands that begin with the word "FORCE" (such as FORCEPROP, FORCEADD, FORCEBUB, FORCENOTE, and FORCEQUIT).

Related Commands

IGnore
Library
Set
Use

Searchstack

Syntax Dialog Box Procedures

Syntax

```
SEArchstack <cr>
```

Description

Accesses the SEARCHSTACK BROWSER form. This form lists the libraries and directories that are currently accessed through the USE or LIBRARY commands.

BROWSE opens or updates the DIRECTORY BROWSER form that lists the contents of a library or directory.

USE places the active directory at the top of the search list and makes it your current working directory. There is no limit to the number of directories and libraries that can be in use at one time.

IGNORE   deletes the specified directory or library from the active search list. When you select a directory or library to ignore, the editor prompts you to be sure you want to ignore it. Move the cursor to the message window, type Y or N, and press <cr>. The specified directory or library is removed from the search stack.

LIBRARY accesses the AVAILABLE LIBRARIES form that lists all the available libraries. From AVAILABLE LIBRARIES, you can select any number of libraries to add to the active search list. As you select libraries, the library names appear in the SEARCHSTACK BROWSER form.

Section

Syntax   Procedure   Related Commands

Syntax

```
SECtion [pin_number]point
```

Description

This command displays different pin numbers for different sections of a symbol. The SECTION command lets you assign physical part sections to selected logical parts. As you step through the different sections of a symbol, the pin numbers of each section are displayed on the drawing. Sectioning a part automatically assigns path properties to the drawing.

If the logical part selected can be assigned to a section, the pin numbers for the section are displayed on the drawing. If the same part is selected again, the next section is selected and the new pin numbers are displayed. This makes it possible to step through all the different possible sections by pointing to the same part.

To assign a specific section directly, enter a pin_number that uniquely defines the section and then point to the part. This avoids having to step through each section individually.

To remove section information from a part, use the REPLACE command to replace the sectioned symbol with a new copy of the part.

You can section only parts with SIZE = 1 or HAS_FIXED_SIZE characteristics. To assign sections to a HAS_FIXED_SIZE part, point to the pin of the section to be assigned. To swap sections within a HAS_FIXED_SIZE part, use the PINSWAP command.

Example

To assign a specific section to an `LS00` component, type `section 11` in the console window and click the component in the design window.

Note: You can click the component again to change the section.

Related Commands

Backannotate
Pinswap
Set

Select

Syntax Procedure Related Commands

Syntax

```
SELect [group_name | DEFault][type...]{selection...| group_name |ALL}
```

where type is:

```
BOdies | PRoperties | NOtes | WIres | DOts
```

and selection is:

```
{Lpoint Lpoint} | Ctrl+Rpoint | Mpoint
```

Description

Provides a stretchable rectangle to specify the boundaries of a group. The group is defined as a collection of objects.

- If the first argument is a single-letter group name, that group will become the current group. If a group name is not specified, or DEFault is specified, a single-letter group name will be automatically assigned to the group that is created.
- The selection can be restricted to a specified type or set of types by providing one or more type arguments.

  For example, the command `select A bodies properties` includes only the components and properties among the objects you have selected for grouping in group

A, even though there are notes, wires or dots among the objects you have selected for grouping.

- You can click the left mouse button, drag the mouse to define the opposite corners of a stretchable rectangle which contains the objects to include in the group, and click again. You can draw additional rectangles to include other objects in the group.
- Press *Ctrl*+right mouse button to include individual objects in the group.
- Click the middle mouse button on another highlighted group to include its contents in the current group.
- Use the name of a previously created group to include its contents in the current group.

  For example, the command `select C A` includes the contents of group A in group C.
- Use ALL to select all objects in the current drawing.

  For example, the command `select A bodies all` includes all the components in the current drawing in group A.

The console window displays the group name and the number of bodies, properties, notes, dots, and wires in the group.

Related Commands

Auto
Find
Group
Include
Exclude

Set

Syntax   Dialog Box

Syntax

```
SET {option | <cr>}
```

Description

This command establishes the default options for Design Entry HDL editor and overrides the project (.cpm) setting. The `SET` commands can be issued during an editing session or placed in the `START_CONCEPTHDL` section of the `.cpm` file.

Note: The `SET` commands in Design Entry HDL are temporary commands valid for the current session only.

SET <cr> accesses the *Design Entry HDL Options* dialog box. You can then use this dialog box to set different options. You can close the *Design Entry HDL Options* dialog box by typing SET SETFORM OFF in the Console Command window.

The SET command options listed below are grouped by the tabs they belong to in the *Design Entry HDL Options* dialog box. Related options (usually opposites) are listed together and are separated by a slash. The default is shown first. For example, in the *save_workspace* command option, ‘*on’* is the default setting.

Most of the command options are Boolean in nature that can either be ‘*on’* or ‘*off’*. The table below provides a single example of setting a Boolean option (*save_workspace*). The other Boolean options are set in a similar way. For all command options that are not Boolean, an example is provided in the following table. In addition, a single example is provided for command options that are similar, for example, changing the default color of objects, such as dot and wire.

SET Command Options	Description
General	
save_workspace [on/off]	Saves window and toolbar settings when you exit Design Entry HDL. **Example**: set save_workspace on
click_to_type [on/off]	Activates a window when you click in it. Otherwise, a window is activated when you move the cursor into it.
autopan [on/off]	Enables panning behavior that lets you move the window over the drawing, rather than move the drawing inside the window.

SET Command Options	Description
rmb [on/off]	Changes the behavior of the right mouse button (RMB). If the option is turned off: Clicking right displays the context (pop-up) menu Pressing Ctrl+RMB causes a command-dependent action. If the option is turned on, this functionality is reversed, where clicking right causes a command-dependent action and pressing Ctrl+RMB displays the context menu.
multiformat_vectors [on/off]	Allows multiple-format signal names in the design. When you select this option: (), <>, and [] are considered special characters that designate a vectored signal. They cannot be used anywhere else in the signal name. The parentheses must be matched correctly and must contain either an integer or a parameter. Colon (:), comma (,) and ampersand (&) are considered special characters that represent concatenation. They cannot be used anywhere else in the signal name. When this option is not selected, ( ) and [] are considered legal characters that have no special meaning and can be used in signal names. They do not designate vectored signals. Only < > can be used to designate vectored signals. Comma (,) and ampersand (&) are considered legal characters that have no special meaning and can be used in signal names. They do not represent concatenation. Only a colon (:) represents concatenation.

SET Command Options	Description
lmb [on/off]	Changes the behavior of the select and drag mouse operation and for running commands with strokes. If the option is turned off: Select an object on the schematic and drag the mouse to move the object. Press the left mouse button and drag the mouse to select multiple objects on the schematic. To exclude components, properties or wires from the selected objects, click the right mouse button and choose Exclude to exclude components, properties or wires from the selected objects. You can now click on one of the selected objects and drag the mouse to move all the selected objects. Press Ctrl or SHIFT and hold down the left mouse button to run commands with strokes. If the option is turned on: Press Ctrl, select an object on the schematic and drag the mouse to move the object. Press Ctrl or SHIFT, hold down the left mouse button and drag the mouse to select multiple objects on the schematic. To exclude components, properties or wires from the selected objects, click the right mouse button and choose Exclude to exclude components, properties or wires from the selected objects. You can now press Ctrl, click on one of the selected objects and drag the mouse to move all the selected objects. Hold down the left mouse button to run commands with strokes.
addform [on/off]	Opens the Add Component dialog box when you enter the add command in the console window and then press Return.
catform [on/off]	Displays the Category View tab by default when you open the Add Component dialog box or when you enter the add command in the console window and then press Return.
editform [on/off]	Activates the View Open dialog box when you enter the edit command in the console window and then press Return.

SET Command Options	Description
libform [on/off]	Activates the Search Stack dialog box when you enter the lib command in the console window and then press Return. If off, the current search stack is displayed.
ppt_browser [on/off]	Automatically opens the Physical Part Filter dialog box when you open the Add Component dialog box or when you enter the add command in the console window and then press Return.
preselect [on/off]	Activates the pre-select mode for Design Entry HDL menus.
default_page_border_name <name>	Specifies the name of the page border. **Example**: set default_page_border_name A SIZE PAGE
default_page_border_version <version>	Specifies the version of the page border symbol. **Example**: set default_page_border_version 1
Paths	
ppt_optionset_path <path>	Specifies the path to the PPT Option Set file that you want Design Entry HDL to use by default. This file stores the default display settings for physical properties in the schematic and in the Physical Part Filter dialog box. **Example**: set ppt_optionset_path C:\MyDocs
attributes_dir <location of attribute files>	Sets the path for the location of attribute files, e.g. the default allegro_net.att is located at $CDS_INST_DIR/tools/fet/concept/attributes. **Example**: set attributes_dir $CDS_INST_DIR/tools/fet/concept/attributes
CAT_path <location of the directory that contains the category files>	Specifies the directory that contains the category (.cat) files used to organize components by category. The component categories are displayed in the Category View tab of the Add Component dialog box. The Add Component dialog box also loads the category files located in the libraries that you use in your project.
Graphics	

SET Command Options	Description
MOVE_Orthog/ MOVE_Direct	Draws wires that you move as Orthogonal or Direct.
ORthog_wire/DIrect_wire	Draws wires that you add as Orthogonal or Direct.
AUTORoute <ON/OFF>	Automatically routes a wire around objects when you move a component in the drawing.
AUTOHeavy <ON/OFF>	Automatically thickens a wire when you attach a bus signal name to it.
tap_add_signal [on/off]	Automatically inserts the specified tap symbol, bits, and wire names of bus taps when you use the tap command.
tap_body <symbol>	Specifies the tap symbol to be used in a schematic. **Example**: set tap_body tap
AUTODot <ON/OFF>	Automatically displays dots at wire connections.
DOTS_Filled/ DOTS_Open	Adds open or filled dots at wire connections.
BOdy_dot_radius <integer>	Adjusts the diameter of dots at wire connections in symbol drawings. **Example**: set Body_dot_radius 25
LOgic_dot_radius <integer>	Adjusts the diameter of dots at wire connections in schematic drawings. **Example**: set Logic_dot_radius 25
Text	
CAPSLOCK_OFF/ CAPSLOCK_ON	Displays text as all caps.
SIZe <real>	Specifies the size of text (property name, property value, signal name, URL, or note) in the plotted schematic. The default value is 0.082 inches. **Example**: set size 1.000
CEnter_justified/ LEft_justified/ RIght_justified	Justifies text Left, Center, or Right.

SET Command Options	Description
user_editor <editor>	Specifies the text editor that Design Entry HDL displays for certain functions.
	Example: set user_editor wordpad
PIn_size <real>	Adjusts the size of the pin number displayed on the schematic to be larger or smaller. The unit is in inches. The pin number size is not related to Text Size you specify in this dialog box.
	Example: set pin_size 1.000
PRop_display <display_type>	Controls the way properties are displayed-Invisible, Name only, Value only, or Both name and value.
	Example: set prop_display Invisible
ROtate <ON/OFF>	Automatically rotates pin numbers that are attached to vertical pins.
STICKY_OFF/ STICKY_ON	Deletes a default property from a schematic when the property has been deleted from a symbol drawing.
AUTOPath <ON/OFF>	Automatically attaches a PATH property to an added part.
pinprop_vis [on/off]	Controls the visibility of symbol pin properties when the symbol/ component is instantiated on the schematic. Invisible does not display the symbol pin properties. Defined by Component makes pin properties visible or not depending on how property visibility is defined on the symbol.
	Example: Let us say that the symbol PQR.SYM.1.1 has the property NO_SWAP_PIN = TRUE attached to a pin on it. If the visibility for this property is set to Name in the symbol pin, and you select Defined by Component in Design Entry HDL, NO_SWAP_PIN will be visible on component PQR after it is instantiated on a schematic in Design Entry HDL. If you select Invisible, NO_SWAP_PIN will not be visible on the schematic, but will be visible in the symbol.
Color	
COLOR_Arc <color>	Changes the default *arc* color.
	Example: set color_arc peach
COLOR_Dot <color>	Changes the default *dot* color.

SET Command Options	Description
COLOR_Wire <color>	Changes the default *wire* color.
COLOR_Note <color>	Changes the default *note* color.
COLOR_Property <color>	Changes the default *property* color.
color_body <color>	Changes the default *body* color.
occ_color <color>	Changes the default *occurrence property* color.
BACKground_color <color>	Change the default color for the drawing area. **Example**: set background_color red
Grid	
DECimal/FRactional/ Metric	Defines the grid type.
default_grid [on/off]	Displays or hides the grid.
default_grid [line/dot]	Displays the grid as Dots or dashed Lines.
default_grid <size> <mult>	Adjusts the grid size to be smaller or larger and displays every *nth* grid line to define where objects can be placed so that pins do not fall off-grid. This ensures the correct connectivity of wires and symbols. **Example**: set default_grid 0.150 6
default_body_grid [on/off]	Displays or hides the grid.
default_body_grid [line/ dot]	Displays the grid as Dots or dashed Lines.
default_body_grid <size> <mult>	Adjusts the grid size to be smaller or larger and displays every *nth* grid line to define where objects can be placed so that pins do not fall off-grid. This ensures the correct connectivity of wires and symbols.
default_doc_grid [on/off]	Displays or hides the grid.
default_doc_grid [line/dot]	Displays the grid as Dots or dashed Lines.

SET Command Options	Description
default_doc_grid <size> <mult>	Adjusts the grid size to be smaller or larger and displays every *nth* grid line to define where objects can be placed so that pins do not fall off-grid. This ensures the correct connectivity of wires and symbols.
Plotting	
wplot_thin_width	Specifies the width of lines used to draw thin wires, boundaries of components, and text on schematics. By default, the single line width is 1.
wplot_thick_width	Specifies the width of lines used to draw buses and thick wires on schematics. By default, the double line width is 10.
WPLOT_ADJust	Specifies the percentage by which to increase or decrease the size of the drawing. Design Entry HDL then plots the drawing on one or more papers of the specified size. For example, if you have a drawing with Cadence A size page border, the percentage specified is 100, and the paper size selected is A4. The Cadence A size page border is bigger in size than A4. So, the schematic is plotted on more than one A4 paper.
WPLOT_FIT_to_page	Adjusts the size of the drawing so that it fits into one page of the specified paper size. For example, you may have a drawing with Cadence A size page border, and the paper size selected is A4. Even though the Cadence A size page border is bigger in size than A4, the schematic is plotted so that it fits on one A4 paper.
Wplot_screen	Plots the portion of the schematic that is displayed on the screen.
Wplot_sheet	Plots an entire page.
WPLOT_PLOTTER	Sets the plotter to the name specified.
Output	
AScii/NOAscii	Saves an ASCII representation of the logic.

SET Command Options	Description
BInary/NOBinary	Saves a binary representation of the logic.
CONFirm_write <ON/OFF>	Provides confirmation about saving the drawing.
DEPendency/NODependency	Saves an ASCII file with dependency information.
Check	
CHECK_On_write <ON/OFF>	Runs a check whenever you save a design. Errors are recorded in cp.mkr and netlister.mkr.
CHECK_Missing_pins <ON/OFF>	Checks for pin properties that are no longer attached to pins.
CHECK_Unconn_wires <ON/OFF>	Checks for unnamed wires connected to only one pin (NC wires) and for named nets not connected to any pins.
CHECK_SHorted_pin <ON/OFF>	Checks for two pins on one component that are connected to the same wire, i.e. two pins shorted together.
CHECK_PARts_at_same_loc <ON/OFF>	Checks for overlaid components.
CHECK_Hidden_wires <ON/OFF>	Checks for wire segments hidden by portions of components.
CHECK_PIN_near_wire_endpt <ON/OFF>	Checks for wires that do not quite contact pins.
CHECK_Arcs_at_same_loc <ON/OFF>	Checks for overlaid arcs.
CHECK_PINS_at_origin <ON/OFF>	Checks for pins at the origin (0,0) in symbol drawings.
CHECK_TWo_wires_at_pins <ON/OFF>	Checks for wires overlapping a component at the pin.
CHECK_SIGNAMES <ON/OFF>	Checks for multiple names attached to the same signal.
CHECK_SIGNAME_in_body <ON/OFF>	Checks for the SIG_NAME property on a pin in a symbol file.

SET Command Options	Description
CHECK_Body_place_holders <ON/OFF>	Checks for placeholder components that appear due to changes in the related library.
CHECK_PRop_place_holders <ON/OFF>	Checks for placeholder properties that appear due to changes in the related library.
CHECK_PACk_sec_type_props<ON/OFF>	Checks for multiple SEC-type properties on an instance.

set NEXTgroup

This command sets the name of the group you want to create with the Find command.

If a group with the same name exists, this command resets the group and allows you to create a new group with the same name. For example, suppose that group A contains 2 properties. If you run the following console commands on a schematic that has two instances of the `ls04` component, the new group A will contain 2 bodies:

```
set nextgroup A
find ls04
```

set HDL_Direct

This command sets options for the `write` console command. When you save a drawing, the `write` command is executed that writes the drawing onto the disk.

When the HDL_Direct option is on/off, the `write` command writes/does not write the following files in the sch_1 view of the schematic:

- ❑ `verilog.v`
- ❑ `vhdl.vhd`
- ❑ `hdldirect.dat`
- ❑ `viewprps.prp`
- ❑ `*.sir`

Note: The `hier_write` and `uprev_write` console commands are not affected by the HDL_Direct option. These commands always write all the files onto the disk.

set sticky_on and set sticky_off

When you place a component on the schematic, the properties specified on the symbol for the component become the default properties for the instance of the component. You cannot delete the default properties for the instance.

If you delete a property or modify the property name on the symbol, the property may still be present on the instance of the component as a default property. These are called dangling properties. Design Entry HDL displays the following error message when it finds dangling properties on the schematic:

```
The default property <property> [with value <value>] is no longer on the body
<symbol_name>. To turn the deleted default properties into non-default properties
type SET STICKY_ON;GET;SET STICKY_OFF.
```

Note: The dangling properties will not be visible on the schematic or in the *Attributes* dialog box.

- If you want the dangling properties to be converted to non-default properties, run the following console commands in the following order:

  ```
  set sticky_on
  get
  set sticky_off
  ```

  The dangling properties become visible on the schematic or in the *Attributes* dialog box. The non-default properties are applicable only to the instance of the component in the schematic. You can delete the non-default properties on the component.

- If you want to delete the dangling properties, do the following:

  a. Run the `set sticky_off` console command.

  b. Run the `write` console command or choose *File – Save* to save the schematic.

Show

Syntax

```
SHow {option |<cr>}
```

Description

This command temporarily displays objects or information on classes of objects. The temporary information is erased when you redraw the screen.

To see all the SHOW options, type `SHOW <cr>` in the console window. The available SHOW options are displayed in the console window. The various SHOW options are described in the table below.

Attachments

Displays a line between visible properties and the property owner

COLor pt

Displays the color of the selected object

The color is displayed in the Design Entry HDL console window and as ticker text in the status bar.

COOrdinate pt

Gives the editor the location of the specified point in coordinates

DISTance pt pt

Displays the distance between two selected points.

- Clicking the left mouse button at two grid points displays the distance between the two grid points.
- Clicking the middle mouse button (in a three button mouse) or *Ctrl*+left mouse button at two points on the screen displays the distance between the two screen points.

The distance is displayed in the Design Entry HDL console window and as ticker text in the status bar.

Group pt

Highlights the group and lists the group name and contents. You can also specify the group by name.

History

Lists all drawings read in during the editor session

Keys

Lists the command string assigned to each function key

Modified

Lists drawings in all viewports that were modified but not saved during the editor session

Net pt

Highlights the selected net and displays the net name. You can also specify the net by name.

The net name is displayed in the Design Entry HDL console window and as ticker text in the status bar.

Origins

Displays an asterisk at each object origin (including text) on display

PIns

Displays an asterisk at each pin on display

PRoperties

Displays the names and values of all properties on the display (including any invisible properties)

PWd	Displays the name of the current project directory The current directory is displayed in the Design Entry HDL console window and as ticker text in the status bar.
RELease	Lists the date and number of the current version of the editor
RETurn	Lists all the drawings that the editor can return to, in the order of return
Size pt	Displays the size of the selected text string (property name, property value, URL, signal name, note) in inches. The text size is displayed in the Design Entry HDL console window and as ticker text in the status bar.
VECtors pt	Displays the pin names of the selected symbol

Example

To display the color of an object, type `show color` and click the object in the design window. The color of the selected object is displayed in the console window.

Signame

Syntax Procedure Related Commands

Syntax

```
SIGname {point signal_name | signal_name point }...
```

Description

This command attaches signal names to wires or pins. There are two ways to attach a signal name.

- Type SIGNAME and point to the location for each signal name. A rectangular box appears at each location. Type the text for the signal name and press <cr>.
- Type SIGNAME and enter one or more signal names. Specify the points to place the signal names on the drawing. The signal name is attached to the wire or pin that is closest to the specified point.

Signal names are handled internally as properties. For example, attaching a signal called BUS ENABLE to a wire is equivalent to attaching a property SIG_NAME=BUS ENABLE, to that wire.

When editing a SYMBOL drawing, signal names are known as PIN_NAME properties. They can be attached only to pin connections.

Related Commands

Busname
Property

Smash

Syntax Procedures Related Commands

Syntax

```
Smash {point | group_name}...
```

Description

This command breaks a symbol into individual wires, arcs, and notes. Any properties attached to the symbol are deleted. The SMASH command works on individual bodies and on groups.

The SMASH command is useful for creating library symbol drawings. For example, once a 2-input AND gate exists, N-input AND gates can be made by using the following commands:

```
edit N AND.body
add 2 AND <pt>
smash <pt>
```

Attach the N inputs and write the drawing. Because 2 AND is no longer a symbol, the editor writes the drawing instead of producing an error message as it does when a symbol is added to a symbol drawing.

Procedures

- Smashing a component

- Breaking up components in a group

Related Commands

Filenote
Scale

Spin

Syntax Related Commands

Syntax

```
SPIn point...
```

Description

This command changes the orientation of text strings and components. Spins are in 90-degree increments (0, 90, 180, and 270). When you spin a symbol, all notes and properties are also spun and translated. You can then act on the properties independently.

The SPIN command does a true rotation of the symbol, as opposed to the ROTATE command, which mirrors for 180 and 270 degrees. Use SPIN with care; allowing 180-degree rotations of devices may reverse the order of the pins (for example, in mergers). This can cause subtle errors in a design.

Related Commands

Mirror
Rotate
Set
Version

Split

Syntax Procedure

Syntax

```
SPLit point point...
```

Description

This command splits a single wire into two wires or separates two or more objects that are placed at the same location. The currently selected item blinks so you know what you have selected.

To split a single wire into two wires, type SPLIT and select a point along the wire with the mouse. If you want to separate the end of a wire from some objects, point near the end of the wire. If you want to break a wire into two attached segments, point near the middle of the wire. This creates a bend in the wire at the selected location. Select and position the appropriate section of the wire.

To disconnect two or more items at the same location, type SPLIT and select a location. One of the objects at that location is attached to the cursor and can be moved on the screen. Select the original location again to separate the second object. Continue to select objects until the correct item is attached to the cursor. You can cycle through the objects repeatedly. Move the object to its new location and click the mouse to place the object.

Strokefile

Syntax Procedure Related Commands

Syntax

```
Strokefile {strokes_file | <cr>}
```

Description

This command loads a user-defined custom *`strokes_file`*. Strokes are user-defined line drawings you create with the mouse. You can customize command entry in the editor by using strokes.

When you access the editor, a set of default strokes is initially read from the file `<`*`your_install_dir`*`>/tools/fet/concept/concept.strokes`.

You can specify a file containing your own strokes by issuing the STROKEFILE command at the keyboard or by including the command in your `START_CONCEPTHDL` section of the `.cpm` file.

To use a stroke, press and hold the left mouse button and draw the required stroke. If the stroke you enter does not match an existing stroke, an error message is generated. If you draw a valid stroke, the related command is executed.

Strokes must be entered in the same direction as they were created. This allows you to have two strokes that look the same but that are bound to different commands. For example, you may have two different strokes that both appear as diagonal lines but are bound to different commands. The difference is that one stroke is drawn from upper left to lower right, and the other is drawn from lower left to upper right. When you use the editor to view strokes, a cross signifies the beginning of the stroke.

STROKEFILE <cr> brings up the Browser. You can then select the *`strokes_file`* from the form.

Related Commands

Loadstrokes

Swap

Syntax   Procedure

Syntax

```
Swap point1 point2...
```

Description

This command swaps two properties or two notes. Only two notes or two properties can be swapped, not a note and a property. Default properties and those generated by the PINSWAP, SECTION, and BACKANNOTATE commands cannot be swapped.

symread

Syntax

```
symread <libname>cellname.sym.<versionnumber>
```

Description

This command updates a symbol from the disk. In the syntax, `libname` is the library name in which the component is present, `cellname` is the name of the component, and version number is the symbol version that you want to update.

System

Syntax

```
System {operating_system_command | <cr> }
```

Description

This command accesses the operating system. To execute a particular system command, enter it on the same line as the SYSTEM command. Without an argument, the editor provides an interactive shell. You are connected to the operating system and can run any operating system commands.

To exit from the operating system and return to the editor, type Ctrl-D.

Tap

Syntax Related Commands

Syntax

```
TAp bus_tap_value point point...
```

Description

This command taps a bit or a set of bits from a bus and wires them to a pin. The bus_tap_value is the bit or set of bits you want to tap off the bus. You can enter any number of bus_tap_value(s) before you start selecting points. There are two modes of this command. In mode 1, you first select the pin you want to tap to and then the bus to tap from. In mode 2, you first choose the bus and then choose the pin.

Mode 1:

1. Enter one or more bus_tap_values -- such as 3..0, 5, 2 etc. separated by CARRIAGE_RETURNs. If you do not enter a bus_tap_value, a question mark is used for the tap bits.
2. Choose the pin you want the tap to go to. Use any mouse button. Now, you have a wire attached to the mouse.
3. If you click at a point with the left mouse button (and the point is not too close to a bus, you will get a kink at the point you clicked and you can keep drawing your wire. If you choose a point very close to a bus, or you use the middle or left mouse buttons, the command will find the closest bus, and draw a wire from your last point to the bus. It also adds a tap symbol, called "CTAP" by default, between your bus and the wire. The BN property on the tap symbol will be given the appropriate value. If the set option TAP_ADD_SIGNAL is ON, a signal name will be attached to the wire, specifying the bus name the tap has gone to and the bits tapped assuming that the bus you tap from has a vectored name.
4. The next signal name you entered will be used for the next tap (i.e. go back to step 1 or 2).

Mode 2:

1. Enter one or more tap names -- such as 3..0, 5, 2 etc. separated by Carriage Returns If you do not enter a tap name, a question mark will be used for the tap bits.
2. Choose the bus you want the tap from. Use any mouse button.
3. Now, you have a wire attached to the mouse. If you click at a point with the left mouse button (and the point is not too close to a pin, you will get a kink at the point you clicked and you can keep drawing your wire. If you choose a point very close to a pin, or you use the middle or right mouse buttons, the command will find the closest pin, and draw a wire from your last point to the pin. It will also add a tap symbol called "CTAP" by default, between the bus you chose and the first wire segment. It will give the BN property on the pin of the CTAP symbol the bits you specified. It will add the signal name to the wire you are adding if you so desire (specify your choice with the set variable TAP_ADD_SIGNAL).

   Note: In this mode, you can terminate the tap wire, by clicking at the same point twice, (with the YELLOW (left) mouse button) after the tap symbol has been added. This is useful if you haven't yet added the component you want the tapped wire to go to, or you want the tapped wire to go to another wire etc.

4. The next signal name you entered, will be used for the next tap (i.e. go back to step 1 or 2).

Advanced User Section

Restrictions: The tap command works with very specific kinds of tap bodies.

Rules for creating a tap symbol are

The tap symbol should

- Have exactly two pins.
- One pin MUST be at the origin of the symbol.
- This pin may NOT have a BN property attached to it.
- The second pin MUST be located on the positive x axis i.e. its coordinates should be (x, 0), where x > 0.
- The second pin MUST be on a grid point.
- The second pin MUST have a BN property.

It is recommended that the first pin have a \NAC \NWC on it and the second pin have a \NAC.

As a default the tap symbol CTAP from the standard library is used. If you want to change the tap symbol, use the command

```
set TAP_BODY <yourtapname>
```

For example,

```
set TAP_BODY ktap
```

Related Commands

Bustap

Textsize

Syntax   Procedures

Syntax

```
textsize <size in inches> <group name>
```

to change text size of properties in a group.

OR

```
textsize <size in inches> Click on property
```

to change text size of a property.

```
textsize <size in inches> Click on note
```

to change the text size of a note.

You can specify a text size that has up to three decimal places. The minimum text size that you can specify is 0.008 inches and the maximum is 1.740 inches. The text size you specify should be a multiple of 0.002 inches.

Note: The above commands change the size of text in the plot.

Procedures

- Setting text size of a property or a note
- Changing size of text in a group

Undo

Syntax Procedure Related Commands

Syntax

```
UNDo <cr>
```

Description

This command undoes the operation of the previous drawing command. The editor keeps a list of operations performed during the current editing session. Repeated applications of UNDO reverses the effects of events according to this list.

You can undo past a write to the beginning of a session if you edit a single drawing, write it, and then continue to edit the original drawing. If you edit a second drawing immediately after writing the first, and then return to the first drawing, you cannot undo past the point where you accessed the second drawing.

`UNDO` only affects the current drawing. For example, to undo the move command used to transfer an object from one drawing to another, type UNDO in the destination drawing to remove the object, and then type UNDO in the original drawing to replace the object.

Note: If you have performed an operation relating to the addition of a component, which has a property whose value exceeds 256 characters, and want to reverse the operation, the `UNDO` command will not work.

Related Commands

Redo

Unhighlight

Syntax Related Commands

Syntax

```
UNHighlight [ Net | PArt | PIn | Any ] pt
UNHighlight [ Net | PArt | PIn | Any ] object_name
```

Net, Part, and Pin are used to specify the object type you want to unhighlight. You may use Any if you want all selected objects to be unhighlighted. This is also the default argument if you do not specify any object type.

There are two ways to select an object:

- pt

  To pick a component to be unhighlighted, point to the object and press the left button. Select a group by pointing to the required group and pressing the middle button.

- object_name

  If you prefer typing in the name of the object, you may use the second version of this command.

For nets, the object name is the signal name of the net, for example, FOO.

For parts, the object name is the value of the PATH property attached to that part or component, for example, 7P.

For pins, the object name is the value of the PATH property attached to that part or component to which the pin belongs, followed by a period ("."), followed by the name of the pin, for example, 7P.A<SIZE-1..0>.

Wildcards such as * and ? may be used in specifying the object name,

for example, FOO*, 7*P, 7P.A*.

Description

The Unhighlight command is used to unhighlight an already highlighted object throughout the system.

Related Commands

Highlight
Dehighlight

Updatesheetvars

Syntax

updatesheetvars

Description

This command updates the custom text variables for page numbers on all pages in a design.

Use

Syntax Related Commands

Syntax

```
Use {library_name | <cr>}
```

Description

This command specifies a working library. Library_name refers to the name of the library you want to use. If the library is not in the current directory, include the pathname.

USE places the specified library at the top of the active search list, and it becomes your current working library. If the library has been previously specified, it is moved to the top of the library search stack.

There is no limit to the number of libraries that can be in use at one time.

USE <cr> brings up the file browser form. The user can then select a library_name from the form.

Related Commands

Directory
IGnore
Library

Vectorize

Syntax

```
VECtorize
```

Description

Note: This command works only on Solaris.

This command creates a file named `vector.dat`, which contains a vector plot format version of the current drawing. This file can be used to transmit files to other machines or drive a pen plotter (with the aid of a format conversion program).

Version

Syntax Procedures Related Commands

Syntax

```
VERsion {point | group_name}...
```

Description

This command selects an alternate version of a symbol. Some bodies are created with different symbolic representations. For example, the NAND gate is equivalent to an INVERT-OR gate by DeMorgan's Theorem. A NOR gate is equivalent to an INVERT-AND gate. All versions of a symbol refer to the same logic drawing.

To step from one representation of a symbol to another, type VERSION and point to a symbol. The editor determines the current version of the symbol and displays the next version in the sequence. Continue pressing the mouse button to cycle through all the symbol versions. After the last version of the sequence is displayed, the first version is redisplayed.

You can use the FIND command to group all occurrences of a specified symbol, and then issue the VERSION command with the group_name option to globally change the drawing. The center button changes the version of the bodies in the group closest to the cursor.

The separate versions of a symbol must all make reference to the same logic drawing. Using a different version of a symbol has no influence on the logic drawing defining it.

Procedures

- Versioning a component
- Versioning components of a group

Related Commands

Add
Replace
Rotate

Vpadd

Syntax

```
VPAdd     <cr>
```

Description

This command creates a new viewport. Viewports let you look at different views of the same drawing or open different drawings at the same time. You can zoom the windows independently to focus on different sections of the design, use the WIRE and ROUTE commands to connect points between viewports, and use the new viewport as a global view of the original design.

When you create a viewport, the current drawing is copied to the new viewport and fit to the size of the window. Any operations you perform in either window appear on both copies of the drawing.

Select the active viewport by placing the cursor within the viewport. If the SET command option CLICK_TO_TYPE is ON, click the left mouse button within the window in order to make the window active.

To edit a different drawing, make sure the cursor is in the correct viewport and use the EDIT command to access the new drawing. When you edit different designs simultaneously, you can use the COPY and MOVE commands to share information between the drawings.

No matter which viewport is active, all commands and system responses appear in the message window of viewport1 (the main viewport). The message window is not considered a part of viewport1. If you move the cursor from one viewport into the message window of viewport1, the system still considers the viewport you were in as the active viewport. To activate viewport1, make sure the cursor enters the graphic window area of the viewport.

To move a viewport, place the cursor in the title bar of the viewport, press and hold the left mouse button, and move the viewport to the new location. To resize a viewport, place the cursor in a corner of the viewport, press and hold the left mouse button, and resize the viewport.

Vpdelete

Syntax Related Commands

Syntax

```
VPDelete point...
```

Description

This command deletes an existing viewport. VPDELETE remains active until you enter a semicolon or select another command.

To delete a viewport, click the left mouse button in the viewport. Any drawings that are active in the viewport are not saved but are noted as being modified drawings. Use the SHOW MODIFIED command to list drawings that have been changed but not written. To save the modified drawings, edit the drawing in another viewport and issue the WRITE command.

Related Commands

Vpadd

Window

Syntax Related Commands

Syntax

```
WINdow {; | Down | Fit | In | Left | Out | Previous | point;|point point; | point
point point |Right  |  scale_factor |Up}...
```

Description

This command changes the view of the current drawing. WINDOW can use up to three arguments. If there are fewer than three arguments, terminate the command with a semicolon.

The command options are

WINDOW	Redraws the image without changing the center or scale. This refreshes the screen and clears all messages.
DOWN	Repositions the center of the screen down below the drawing (moves the drawing up on the screen).

FIT Fits the drawing to the entire screen.

IN Enlarges the size of the drawing on the screen.

LEFT Repositions the center of the screen to the left of the drawing (moves the drawing right on the screen).

OUT Reduces the size of the drawing on the screen.

PREVIOUS Switches from the current window scale and positions to the previous window scale and position.

Pt ; Pans the drawing and makes the point the center of a new screen display of the drawing. The scaling of the drawing remains the same. Use the right button to enter the single point.

Pt pt ; Defines a rectangle with the specified points at opposite corners. The rectangle expands to fill the screen, providing a close-up view of the specified portion of the drawing.

Pt pt pt: Changes the size of the drawing depending on the ratio between the points. The first point defines the new center of the drawing, and the display becomes either larger or smaller, depending on the other points. The drawing is redisplayed with the first point at the center. If the distance between the first point and the third point is greater than the distance between the first point and the second point, the items appear larger; if the distance is smaller, items appear smaller.

RIGHT Repositions the center of the screen to the right of the drawing (move the drawing left on the screen).

Scale_factor Enlarges or reduces the drawing by the specified factor. The center of the window remains the same.

UP Repositions the center of the screen above the drawing (move the drawing down on the screen).

Related Commands

Zoom

Wire

Syntax Procedure Related Commands

Syntax

```
WIRe {[signal_name] point point}...
```

Description

This command adds wires to a drawing. The wire begins at the first point specified and runs to the second. Specify additional points to draw a wire with more segments. To snap the wire to the nearest vertex, press the right button.

To end a wire at a pin, dot, or other wire, press the left button.

To end a wire in a free space, press the left button twice at the final point.

Because schematics almost exclusively use orthogonal wires, the default wire mode is orthogonal (bent). Once the wire is started and the cursor changes direction, the attached wire remains orthogonal, whether the cursor is moved horizontally, vertically, or diagonally. To bend a wire, press the left button. Press the center button to change the orientation of the bend. If you press the center button a second time, the wire becomes diagonal. A third press returns the wire to the first orthogonal position.

If you enter a signal_name, the wire will be given that signal name.

See the BWIRE, BROUTE, ROUTE commands. See also the SET and SHOW commands to change default wiring behavior.

Related Commands

Broute
Bwire
Route

Write

Syntax Procedure Related Commands

Syntax

```
WRite [<directory>][drawing_name][.[type][.[version][.[page]]]] | <cr>
```

Description

This command writes the current drawing onto the disk. WRITE always asks for confirmation to write a drawing, even if no errors exist in the drawing. This is a safety feature to prevent unintentional writes. <DIRECTORY> is the directory where the drawing resides. If no directory is given, the drawing is written to the directory from which it was retrieved. If you write a newly-created drawing without giving a directory name, the drawing is written to the current directory. If only a directory name is given, the drawing is written into the specified directory. This is useful when copying drawings between directories.

DRAWING_NAME is the name of the drawing to write. If no drawing_name is specified (WRITE <cr>), the drawing is written to the file name shown on the status line at the top of the display. If no drawing type, version number, or page number are specified, the default is SCH.1.1.

If you enter a drawing name and a drawing with that name already exists in a directory, a warning message is displayed. Type YES to overwrite the existing drawing with the new drawing. Type NO to cancel the write.

Related Commands

Diagram
Edit
Get

Zoom

Syntax Procedures Related Commands

Syntax

```
Zoom{;| Down |Fit |In | Left | Out | Previous | point; | point point; | point point
point | Right |  scale_factor | Up }...
```

Description

This command reduces and enlarges portions of the drawing on the screen. ZOOM can use up to three arguments. If there are fewer than three arguments, terminate the command with a semicolon. The command options are:

ZOOM Redraws the image without changing the center or scale. This refreshes the screen and clears all messages.

FIT Fits the drawing to the entire screen.

LEFT Repositions the center of the screen to the left of the drawing (moves the drawing right on the screen).

PREVIOUS Switches from the current window scale and position to the previous window scale and position.

Pt pt Defines a rectangle with the specified points at opposite corners. The rectangle expands to fill the screen, providing a close view of the specified portion of the drawing.

RIGHT Repositions the center of the screen to the right of the drawing (moves the drawing left on the screen).

U Repositions the center of the screen up above the drawing (moves the drawing down on the screen).

Scale_factor Enlarges or reduces the drawing by the specified amount. The center of the window remains the same.

Pt pt Changes the size of the drawing depending on the ratio between the points. The first point defines the new center of the drawing, and the display becomes either larger or smaller, depending on the other points. The drawing is redisplayed with the first point at the center. If the distance between the first point and the third point is greater than the distance between the first point and the second point, the items appear larger. If the distance is smaller, items appear smaller.

Pt Pans the drawing and make that point the center of a new screen display of the drawing. The scaling of the drawing remains the same. Use the right button to enter the single point.

OUT Reduces the size of the drawing on the screen.

IN Enlarges the size of the drawing on the screen.

DOWN Repositions the center of the screen down below the drawing (moves the drawing up on the screen).

Procedures

- Zooming in and out of a drawing
- Panning the drawing

Related Commands

Window

2

Nongraphical Design Entry HDL (nconcepthdl)

The graphical Design Entry HDL:

- Only runs on graphics workstations
- Always draws graphics on the screen
- Only runs in the background under the X-Windows system

`nconcepthdl` is a version of Design Entry HDL that allows you to run Design Entry HDL scripts in a nongraphical mode. This allows you to run Design Entry HDL

- Without a graphics terminal
- In the background

`nconcepthdl` is useful for running a large batch processes, such as `hardcopy` or `backannotate`, without having to invoke Design Entry HDL.

Running nconcepthdl

To run `nconcepthdl`, use the following syntax:

```
nconcepthdl
     -proj <project file>.cpm
     [-scr <additional script filepath relative to the project directory>]
     [-log <log filepath>]
     [-ignoreprojscr]
```

`nconcepthdl` first executes the Design Entry HDL script specified in the `<project file>.cpm` file, if the script exists. Then if the `-scr` option is also specified, that script is executed. If you use the `-ignoreprojscr` option, the Design Entry HDL script specified in the `<project file>.cpm` file will not be executed.

If the `-log` option is not specified, the messages are thrown to `stderr` as well as the `nconcept.log` file in the project's temporary directory.

`nconcepthdl` supports all Design Entry HDL console commands except the following commands:

- GOTOSHEET
- Windows plotting (`plot`)
- Updatesheetvars

For more information on the Design Entry HDL console commands, see Appendix 1, "Console Command Reference."

3

Using the Standard Library Symbols

The Standard Library is a Design Entry HDL library of symbols that are useful for manipulating signals in a structured design, applying properties to an entire design, and documenting information on the schematic. All the symbols in the Standard Library are supported by Design Entry HDL and are translated correctly into VHDL and Verilog text descriptions.

The Standard library is included in the default list of project libraries for all projects.

VHDL_DECS and VERILOG_DECS Symbols

The Standard Library includes two declaration symbols, VHDL_DECS and VERILOG_DECS. These symbols are not required on schematics, but they are useful for adding Verilog and VHDL-related properties that are applicable to the entire schematic. For example, to set the VHDL logic type for all vectored ports and signals in your drawing, you can add a VHDL_DECS symbol to the first page of the schematic and attach a VHDL_VECTOR_TYPE property to it.

Use the VHDL_DECS symbol for VHDL designs and the VERILOG_DECS symbol for Verilog designs. The two symbols are similar.

Follow these rules if you use a VHDL_DECS or VERILOG_DECS symbol:

- Use either symbol, VHDL_DECS or VERILOG_DECS, but not both, on a schematic.
- Place the symbol on the first page of the schematic.

Properties on VHDL_DECS and VERILOG_DECS

You can use the following properties on a VHDL_DECS or VERILOG_DECS symbol. The default values of all these properties, except VHDL_GENERICxx, VLOG_PARAM, and /PARAM, are specified in the *Output* tab of the *Design Entry HDL Options* dialog box and apply to the entire design. By using these properties on a VHDL_DECS or VERILOG_DECS symbol, you can override the defaults for individual drawings.

Note: In release 16.5, any property changes to a drawing body do not get reflected across the complete design/block. Properties are not applied on drawing bodies and therefore are not updated on every instance of the design/block.

Properties used when only VHDL is generated from a Design Entry HDL design

- VHDL_GENERICxx
- VHDL_SCALAR_TYPE
- VHDL_VECTOR_TYPE
- LIBRARYn
- USEn

Customizing the VHDL_DECS or VERILOG_DECS Symbol

If the VHDL_DECS and VERILOG_DECS symbols are not suitable for your project, you can customize them by either creating a new symbol or by editing a copy of the symbol. You can also use a page border as a declarations symbol.

To create a new declarations symbol

1. Create a new symbol.
2. Copy all visible and invisible properties from the VHDL_DECS or VERILOG_DECS symbol in the Standard Library to the new declarations symbol and define their values.

To edit a VHDL_DECS or VERILOG_DECS symbol

1. Copy the VHDL_DECS or VERILOG_DECS symbol to a new cell.
2. Edit the symbol.

Note: The declarations symbol must have the HDL_SCHEMATIC = TYPE1 property. Do not change its value.

Using a Page Border as a Declarations Symbol

If you do not want to add a VHDL_DECS or VERILOG_DECS declarations symbol to your schematic, you can use a page border that functions as a declarations symbol.

To use a page border as a declarations symbol

1. Add a page border from the Standard Library to your schematic.
2. Remove the COMMENT_BODY = TRUE property from the page border.
3. Copy all visible and invisible properties from a VHDL_DECS or VERILOG_DECS symbol to the page border and define their new values.

Note: Because VHDL_DECS or VERILOG_DECS properties can appear on only the first page of a multiple-page schematic, you must use a different page border for the other pages. The page border for the other pages must have the COMMENT_BODY = TRUE property.

SYNOP_DEC Symbol

Do not use the SYNOP_DEC symbol, an old symbol still included in the Standard Library. Use a VHDL_DECS or VERILOG_DECS symbol instead. For more information, see VHDL_DECS and VERILOG_DECS Symbols on page 151.

The SYNOP_DEC symbol was used for compatibility with Synopsys tools. It is similar to the DECLARATIONS symbol in the Standard library, except for the following:

- The SYNOP_DEC symbol does not have the USER = WORK.ALL property.
- The SYNOP_DEC symbol includes a SYNOPSYS_PRAGMA property to support non-integer generic parameters. Because Synopsys tools do not support generic parameters that are not integers, Design Entry HDL adds the following lines around the information about non-integer generic parameters in the entity and architecture files:

```
pragma translate_off
pragma translate_on
```

Note: If you use the SYNOP_DEC symbol, the VHDL netlist generated by Design Entry HDL does not adhere strictly to the VHDL language guidelines. While elaborating the design, use the Leapfrog compatibility option (`-c`) so that the Leapfrog simulator will accept non-conforming VHDL syntax.

DECLARATIONS Symbol

Do not use the DECLARATIONS symbol, an old symbol still included in the Standard library. Use a VHDL_DECS or VERILOG_DECS symbol instead. For more information, see VHDL_DECS and VERILOG_DECS Symbols on page 151.

HDL_DECS Symbol

Do not use the HDL_DECS symbol, an old symbol still included in the Standard Library. Use a VHDL_DECS or VERILOG_DECS symbol instead. For more information, see VHDL_DECS and VERILOG_DECS Symbols on page 151.

TAP Symbols

Use tap symbols to “tap” or extract a single bit or a range of bits from a vectored signal (bus).

The Standard library has the following tap symbols:

- TAP
- CTAP
- BIT TAP
- LSBTAP
- MSBTAP

All these symbols, except CTAP, have many versions, depending on the rotation of the symbol.

To use a TAP symbol, do one of the following:

- Use the *Component – Add* command to add one of the tap symbols contained in the Standard library. Select the version of the symbol while adding it or after you place it on the schematic.
- Use the *Wire – Bus Tap* command. You can choose the default tap symbol for the *Wire – Bus Tap* command by specifying it in *Graphics* tab of the *Design Entry HDL Options* dialog box.

TAP

Use a TAP symbol to tap or extract a single bit from a bus. The BN property on the TAP symbol determines which bit is tapped. Set the value of this property to the actual bit number you want to tap. The BN property does not have a default value; you must specify its value.

Note: TAP symbols are similar to CTAP symbols. The only difference between them is their graphical representation. However, you cannot use the TAP symbol to auto-tap a bus. If you want to use a symbol other than CTAP to auto-tap a bus, you can create a custom symbol in your local library. The custom symbol must conform to the following:

- It must have two pins.
- The first pin must be at the origin of the symbol and must not have a BN property.
- The second pin must,
  - ❑ be on the grid.
  - ❑ be on the x axis and have a positive x coordinate.
  - ❑ have an associated BN property.
- Both pins must have the AC property and the second pin must also have the WC property.

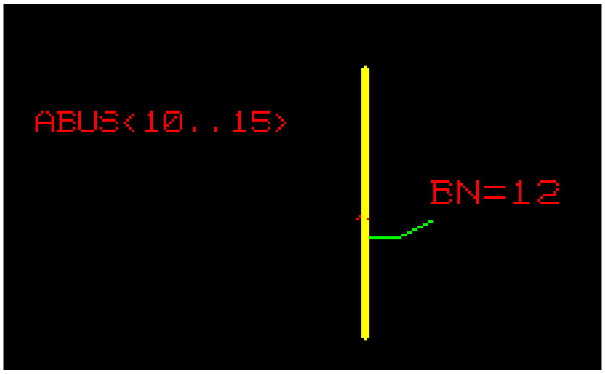
To tap a bit with a TAP symbol

1. Attach a TAP symbol to the bus.
2. Use the *Text – Change* command to set the value of the `BN` property to the bit number that you want to tap.

Example

To select the twelfth bit of the bus `ABUS<10..15>`,

1. Attach a TAP symbol to the bus.
2. Set the value of the `BN` property on the TAP symbol to `12`.



Guidelines for Creating Tap Symbols

Apply these rules to any tap symbols that you create:

1. The tap symbol must have exactly two pins.
2. One pin *must* be at the origin of the tap symbol.
3. The second pin must be on a grid point.
4. The second pin must be located on the x-axis, and x must be > 1.

Attach a BN property to the second pin (*not* to the first pin).

CTAP

Use a CTAP symbol to “tap” or extract a single bit from a bus. The `BN` property on the CTAP symbol determines which bit is tapped. Set the value of this property to the actual bit number you want to tap. The `BN` property does not have a default value; you must specify its value.

Note: CTAP symbols are similar to TAP symbols. The only difference between them is their graphical representation.

To tap a bit with a CTAP symbol

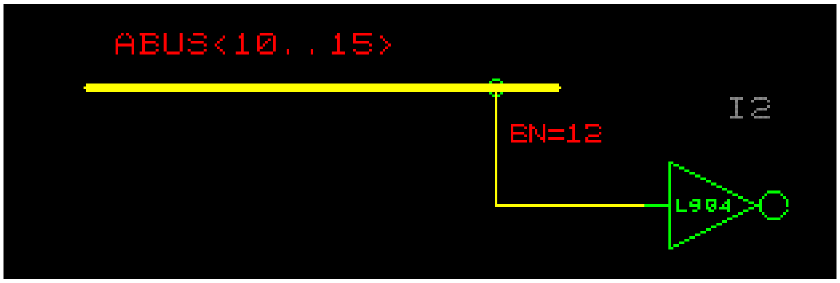
1. Attach a CTAP symbol to the bus.
2. Use the *Text – Change* command to set the value of the `BN` property to the actual bit number that you want to tap.

Example

To select the twelfth bit of the bus `ABUS<10..15>`,

1. Attach a CTAP symbol to the bus.

2. Set the value of the `BN` property on the CTAP symbol to `12`.



BIT TAP

Use the BIT TAP symbol to "tap" or extract a single bit from a bus. The `BIT` property on the BIT TAP symbol determines which bit is tapped. The value of this property is relative, beginning from the Least Significant Bit (LSB). For example, for a bus `addr<10..15>`, if `BIT = 2` on the BIT TAP symbol the twelfth bit of the bus is tapped. The default value of the `BIT` property on BIT TAP symbols is `0`.

To tap a bit with a BIT TAP symbol

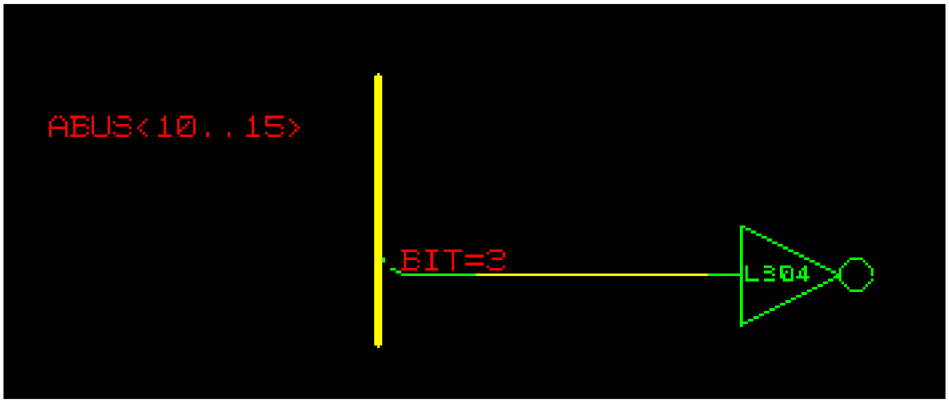
1. Attach a BIT TAP symbol to the bus.
2. Use the *Text – Change* command to set the value of the `BIT` property on the symbol to the bit you want to tap. This value must be between `0` and `<bus_size>-1`.

Example

To select the twelfth bit of the bus `ABUS<10..15>`,

1. Attach a BIT TAP property to the bus.

2. Set the value of the `BIT` property on the BIT TAP symbol to `2`.



LSBTAP Symbol

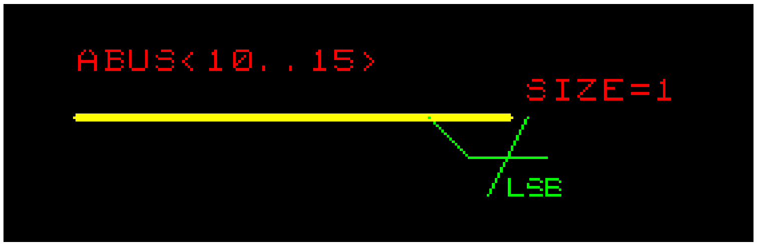
Use the LSBTAP symbol to “tap” or select the Least Significant Bit (LSB) of a bus, or a range of bits beginning with the LSB. The `SIZE` property on the LSBTAP symbol determines which bits are tapped. The default value of this property is `1`, which means that the LSB is tapped.

To tap a single bit

➤ Attach an LSBTAP symbol to the bus. The LSB is tapped.

Example: Using LSBTAP to tap a single bit

To tap the LSB of a bus `ABUS<10..15>`, attach an LSBTAP symbol to the bus. The default value of the `SIZE` property on the LSBTAP symbol is `1`. The tenth bit of the bus is tapped.



To tap a range of bits

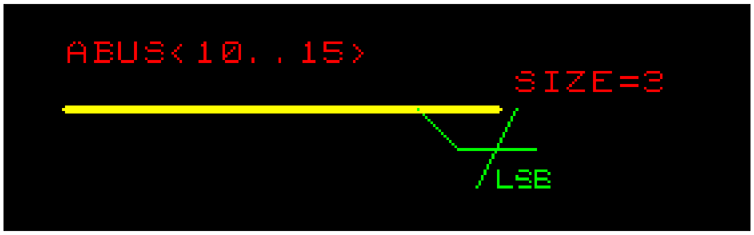
1. Attach an LSBTAP symbol to the bus.

2. Use the *Text – Change* command to set the value of the `SIZE` property to the number of bits you want to tap.

Example: Using LSBTAP to tap a range of bits

To tap bits 10, 11, and 12 from a bus `ABUS<10..15>`,

1. Attach an LSBTAP symbol to the bus.
2. Set the value of the `SIZE` property on the LSBTAP symbol to `3`.



MSBTAP Symbol

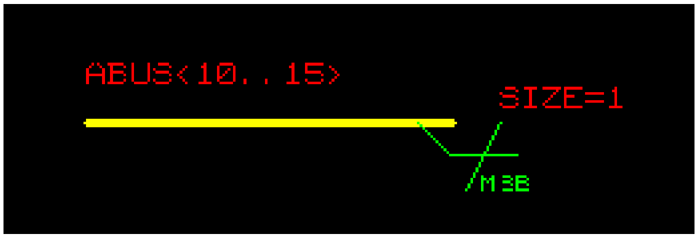
Use the MSBTAP symbol to “tap” or select the Most Significant Bit (MSB) of a bus, or a range of bits beginning with the MSB. The `SIZE` property on the MSBTAP symbol determines which bits are tapped. The default value of this property is `1`, which means the MSB is tapped.

To tap a single bit

➤ Attach an MSBTAP symbol to the bus. The MSB is tapped.

Example: Using MSBTAP to tap a single bit

To tap the MSB of bus `ABUS<10..15>`, attach an MSBTAP symbol to the bus. The default value of the `SIZE` property on the MSBTAP symbol is `1`. The fifteenth bit is tapped.



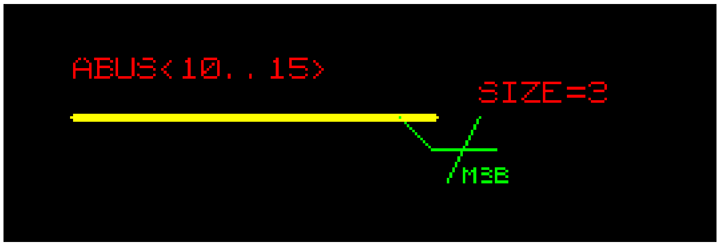
To tap a range of bits

1. Attach an MSBTAP symbol to the bus.
2. Use the *Text – Change* command to set the value of the `SIZE` property to the number of bits you want to tap.

Example: Using MSBTAP to tap a range of bits

To tap bits 13, 14, and 15 from a bus `ABUS<10..15>`,

1. Attach an MSBTAP symbol to the bus.
2. Set the value of the `SIZE` property on the MSBTAP symbol to `3`.



CONCAT Symbols

Use a CONCAT symbol when you want to merge a number of signals, ports, or signal aliases into a group. You can then route this group to a port or instance with a single wire.

For more information on using CONCAT symbols to concatenate signals, ports, or signal aliases, see Signal Concatenation in *Allegro Design Entry HDL User Guide*.

There are nine CONCAT symbols in the Standard library: CONCAT2, CONCAT3, CONCAT4, CONCAT5, CONCAT6, CONCAT7, CONCAT8, CONCAT9, CONCAT10. Use CONCAT2 to merge two signals, CONCAT3 to merge three signals, CONCAT4 to merge four signals, and so on. Each CONCAT symbol has a small note at the top of the symbol indicating the left pin and a small note at the bottom of the symbol indicating the right pin. You will have to zoom in to see these notes.

Concatenated signals can be separated back into individual signals with a MERGE or TAP symbol. (Design Entry HDL works faster if you use one of the TAP symbols instead of a MERGE to slice signals.)

Concatenated symbols are unrelated; concatenation is merely a shorthand notation for signals that run together.

Rules for Using CONCAT Symbols

- The sum of the input signal width must match the width of the output signal connected to the CONCAT symbol. If the input signal or output signal is not named, Design Entry HDL calculates the signal width automatically.
- If the widths of the input signals are specified, and the output signal is not named, the width of the output signal is assumed to be the sum of the widths of the input signals.
- The output of a concatenation can be connected to the input of a CONCAT symbol.
- The output of a slice can be connected to the input of a CONCAT symbol.
- When you create or edit a CONCAT symbol ensure that:
  - The name of the output pin is the highest alphanumeric value of all the pins on the symbol. For example, if the input pins are named `AA`, `DD`, and `FF`, the output pin cannot be named `BB`.
  - The `VHDL_CONCAT=TRUE` property is attached to the CONCAT symbol.

SYNONYM

The SYNONYM symbol is used to specify another name for a signal. SYNONYM symbols are useful for creating locally meaningful names for signals that are spread throughout a design.

Do not use SYNONYM symbols if you want to generate VHDL text for the schematic. Use ALIAS symbols instead. SYNONYM symbols are similar to ALIAS symbols, but Design Entry HDL does not have extensive VHDL checks for SYNONYM symbols. If you use SYNONYM symbols, Design Entry HDL will not detect all the VHDL-related errors and your VHDL output will be inaccurate. Therefore, if you currently have SYNONYM symbols in a design for which you will generate VHDL text, replace them with ALIAS symbols. If you do not intend to generate VHDL text from your schematic, you can use either ALIAS symbols or SYNONYM symbols.

For more information on the ALIAS symbol, see Creating an alias for a signal.

To create a SYNONYM for a signal

1. In Design Entry HDL, choose *Component – Add*.

   Part Information Manager appears.

2. Select `standard` from the *Library* list in the search pane.

3. In the *Cells List*, select *SYNONYM*.

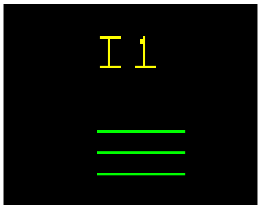
   The SYNONYM symbol gets attached to the cursor.

4. Click in the Design Entry HDL drawing area to place the symbol.
5. Attach the signal for which you want to create a synonym to the left pin of the SYNONYM symbol.
6. Attach the SYNONYM name to the right pin.

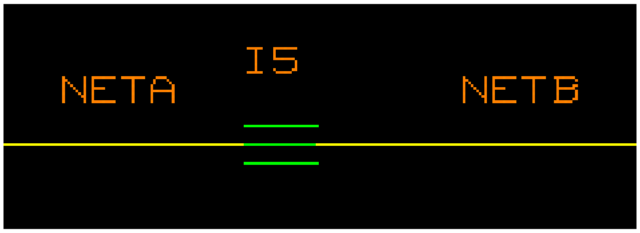
Example

To create a synonym, `NETB`, for `NETA`, do the following:

1. Add a SYNONYM symbol from the Standard Library.



2. Attach a wire with the `NETA` signal name to the left pin of the SYNONYM symbol.
3. Attach a wire to the right pin of the SYNONYM symbol and name it `NETB`.



This creates the following Verilog declaration:

```
alias_bit alias_inst1 (netb, neta);
```

Rules for using SYNONYM symbols

You must follow these rules when using SYNONYM symbols:

- Connect signals of the same assertion: both must be high, or both must be low.
- Connect signals of the same width.

- Synonym symbols cannot be used if you want to generate VHDL text for the schematic. Use ALIAS symbols instead.
- Do not attach the COMMENT_BODY property to a SYNONYM symbol. Allegro Design Entry HDL ignores components that have COMMENT_BODY properties attached to them and omits them from the netlist. As a result, nets with attached synonym symbols will not be shorted or aliased.

For more information on the ALIAS symbol, see Creating an alias for a signal.

TIE

The TIE symbol is similar to the SYNONYM symbol. The difference is that the order in which nets are attached to a symbol is important in case of the TIE symbol, while the SYNONYM symbol is used regardless of the order.

Example

To create a synonym `NET1 for NET2` on two different pages using the TIE symbol,

➤ On page 1, connect Net1 to pin A of the TIE symbol and Net2 to pin B of the TIE symbol.

➤ On page 2, connect Net 2 to pin A of the TIE symbol and Net1 to pin B of the TIE symbol.

An error will be generated in this case as the order is not maintained.

PAGE Borders

The Standard Library includes several page borders that you can use in your schematic. These provide a convenient way of documenting information such as the date, the design name, the page number, the engineer’s name, and the company logo on the schematic.

The following page borders are in the Standard Library:

A Size Page	The A SIZE PAGE symbol is an 8 1/2 x 11 inch border.
B Size Page	The B SIZE PAGE symbol is a 11 x 17 inch border.
C Size Page	The C SIZE PAGE symbol is a 17 x 22 inch border.
D Size Page	The D SIZE PAGE symbol is a 22 x 34 inch border.
E Size Page	The E SIZE PAGE symbol is a 34 x 44 inch border.

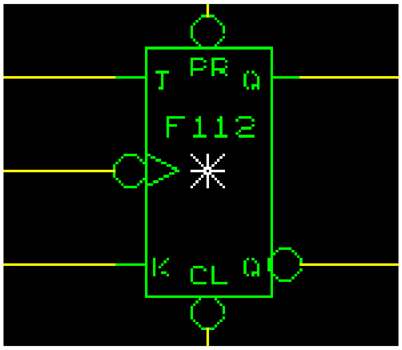
F Size Page	The F SIZE PAGE symbol is a 44 x 68 inch border.

For information on creating and using page borders, refer Creating a page border Symbol.

ORIGIN

The ORIGIN symbol is a pseudo symbol that is required in all symbols. It provides an attachment location for symbol properties and defines the “origin” of the symbol. It is a pseudo symbol because it is the only symbol allowed within another symbol. It is not instantiated and, therefore, not visible in a schematic.

Design Entry HDL automatically uses the ORIGIN symbol to indicate the origin of any symbol. You do not add this symbol manually to a drawing. When you create a new symbol drawing, the ORIGIN symbol appears in the center of the screen.



To view the ORIGIN symbol

➤ In Design Entry HDL, choose *Display – Origins*.

DRAWING

Use DRAWING symbols to attach properties to all instances in the schematic.

If the drawing symbol is instantiated in the schematic and the property `ABC=EFG` is attached to it, all instances within the schematic will get this property. If any particular instance has this property with a different value, then the new value is applicable for the instance.

By default, the DRAWING symbol has the `LAST_MODIFIED` property. The value of the property is the date and time the schematic was last updated. This is set automatically. Note that only one DRAWING object can be used per block.

Examples of other properties that can be attached to the DRAWING body are the `TITLE` and `ABBREV` properties. The `TITLE` property specifies the title of the drawing and must match the schematic name. The `ABBREV` property specifies an abbreviation of the drawing name.

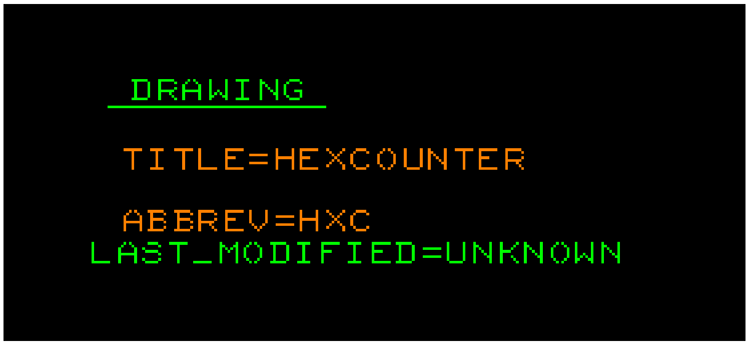
To use a DRAWING symbol

1. Add the symbol to the schematic.
2. Attach the properties to the symbol.

Note: Text macros cannot be used in the value of a `TITLE` property.

Example

The following drawing symbol has the `TITLE` and `ABBREV` properties in addition to the `LAST_MODIFIED` property.



DRAWING

TITLE=HEXCOUNTER

ABBREV=HXC
LAST_MODIFIED=UNKNOWN

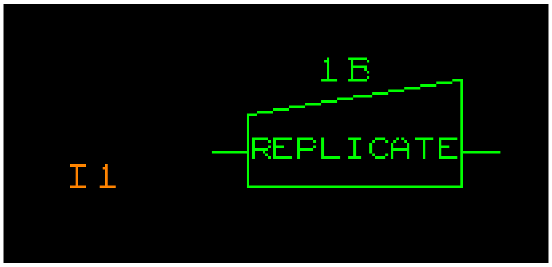
Note: DRAWING bodies should not define any Key or Injected properties. These properties must only be applied from the part table files.

REPLICATE

REPLICATE symbols are usually not added to schematics. They are used by library developers to make models for sizable parts.

Note: You cannot connect a vector net to the input of a REPLICATE symbol.

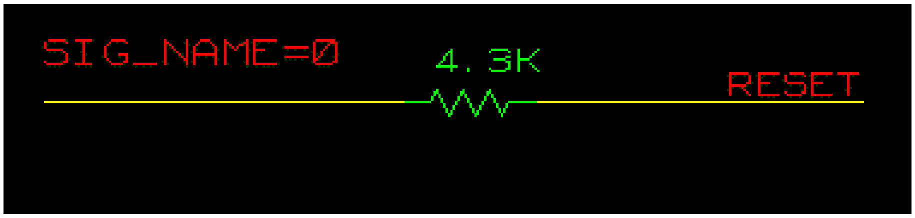
Example



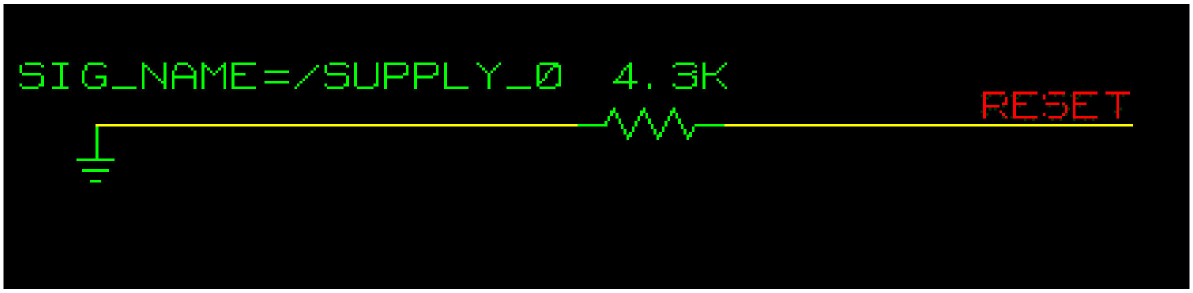
SUPPLY_0

Signals can be defined for the supply0 type in the following ways:

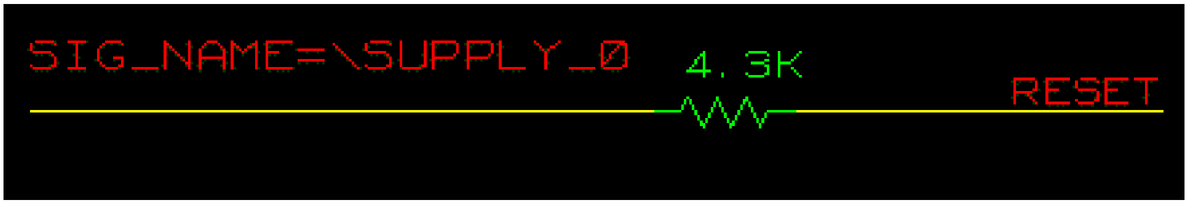
- Use the SUPPLY_0 symbol. The SUPPLY_0 symbol represents a global power signal. You can use this symbol instead of the SUPPLY_0 signal name. In this case, signal supply_0 is of the type supply0.



- Use the `/SUPPLY_0` signal name on any wire. In this case, the signal supply_0 is of the type supply0.

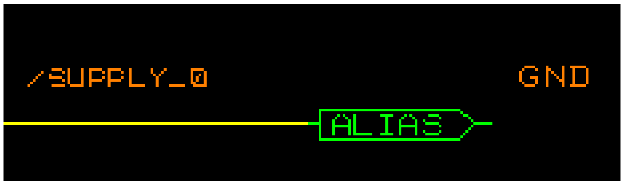


- Name the wire `0`. (Do not put a / in front of 0). In this case, signal 0 will be of type supply0.



- Use the ALIAS symbol to rename the `/SUPPLY_0` signal in the first page of the schematic and then use the alias in other pages. In Figure 3-1, the ground signal `\SUPPLY_0` is given the alias GND. You can use the signal name GND in other areas of the schematic to represent a ground signal.

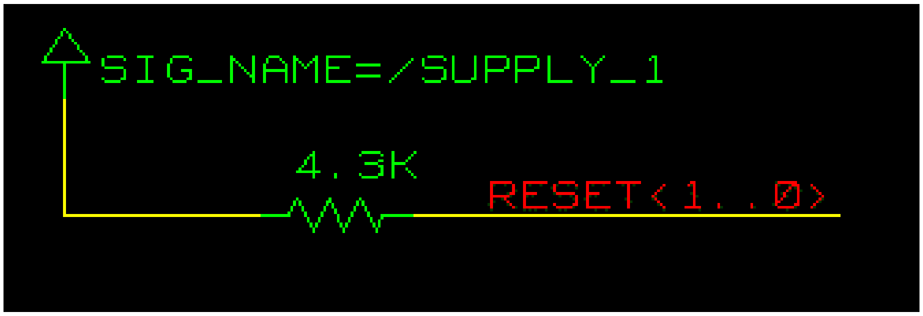
Figure 3-1 Renaming a Global Ground Signal



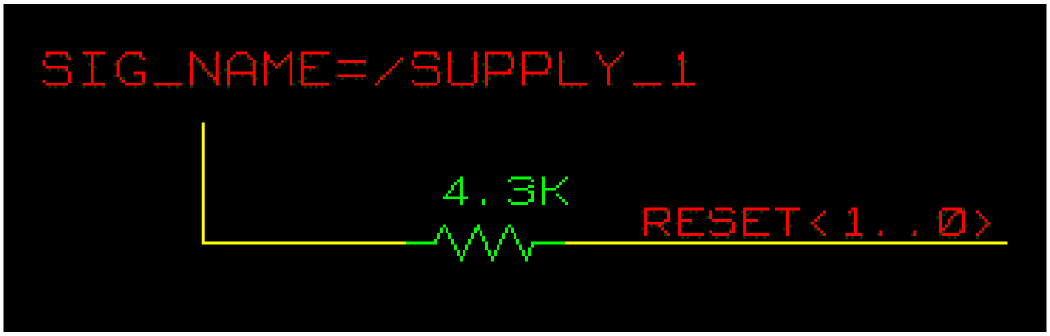
SUPPLY_1

Signals can be defined for the supply1 type in the following ways:

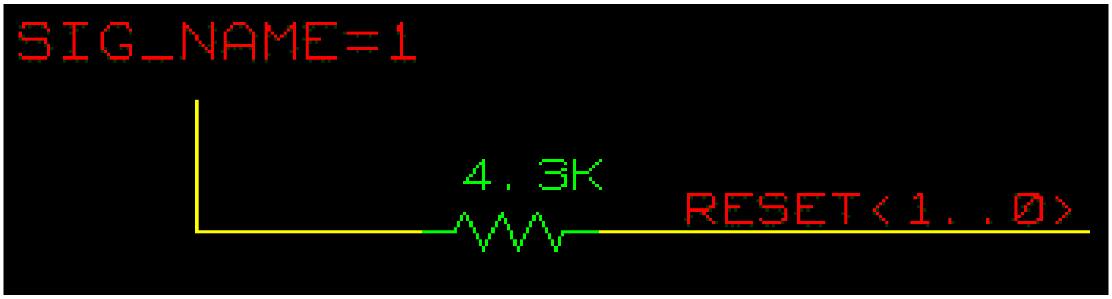
- Use the SUPPLY_1 symbol. The SUPPLY_1 symbol represents a global power signal. You can use this symbol instead of the SUPPLY_1 signal name. In this case, the signal supply_1 is of the type supply1.



- Use the `/SUPPLY_1` signal name on any wire. In this case, signal “supply_1” will be of the type supply1.

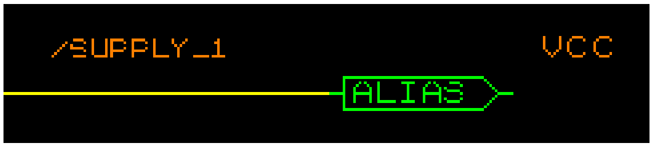


- Name the wire `1`. (Do not put a / in front of 1). In this case, signal 1 will be of the type supply1.



- Use the ALIAS symbol to rename the `/SUPPLY_1` signal in the first page of the schematic and then use the alias in other pages. In Figure 3-2, the power signal `\SUPPLY_1` is given the alias VCC. You can use the signal name VCC in other areas of the schematic to represent a power signal.

Figure 3-2 Renaming a Global Power Signal



PIN NAMES

PIN NAMES symbols are used for hierarchical designs and library development. When you create a hierarchical schematic-symbol drawing pair, use the PIN NAMES symbol to transfer the `PIN_NAME` properties from the symbol drawing to its corresponding schematic drawing. Using the PIN_NAMES symbol eliminates the need to retype signal names and reduces errors in labeling signals and properties.

To use the PIN NAMES symbol

1. Create the symbol drawing.
2. Add pin names to the symbol with the *Wire – Signal Name* command. A `PIN_NAME` property is attached to each of the pins you name.
3. Save the symbol drawing.
4. Create the corresponding schematic drawing. The schematic drawing must have the same name as the symbol drawing, but with a `.sch` extension.

   For example, if the symbol drawing is `CLOCK.SYM.1.1`, type the following in the Design Entry HDL console window:

   ```
   edit clock
   ```

   The `CLOCK.SCH.1.1` drawing will contain the logic that the symbol represents. Place all the required parts and attach wires as required.
5. Add the PIN NAMES symbol from the `standard` library to a corner of the schematic.

   You can also add the PIN NAMES symbol on the schematic using the `pinnames` Design Entry HDL console window command.

For more information, see Pinnames on page 99.

1. Run the `check` console window command.

   Design Entry HDL automatically attaches the names of the pins on the corresponding symbol drawing to the PIN NAMES symbol you added and appends a `\I` suffix (scope = interface) to each signal name. Each pin name is identified with a `SIG_NAME` property.

   Do one of the following if you want to view the signal and property names:

   ❑ Place the cursor on a pin name attached to the PIN NAMES symbol

   ❑ Choose *Text – Attributes* and click on the PIN NAMES symbol to view the signal and property names in the *Attributes* dialog box.

2. Choose *Text – Reattach* to reattach the individual signal names from the PIN NAMES symbol to the appropriate signals on the schematic drawing.
3. Choose *Display – Attachments* to ensure that the signal names have been reattached to the appropriate signals.
4. For drawing clarity, choose *Edit – Move* to relocate the signal names near the associated signals.
5. Delete the PIN NAMES symbol.

FLAG

FLAG symbols are attached to indicate interface signals in a design. FLAG symbols are similar to the PORT symbols in the Standard library. It is recommended to use PORT symbols instead of FLAG symbols.

For more information on PORT symbols, see Adding Ports.

The FLAG symbol has 12 versions:

- Versions 1-4: Input
- Versions 5-8: Output
- Versions 9-12: In/Out

NOT

Do not use the NOT symbol, an old symbol still contained in the Standard library. The symbol was used only to support the Bubble Checker feature of the SCALD compiler, which verified that signals and pins were connected to signals and pins of the same assertion. The NOT symbol was used to avoid this restriction and provided a way of connecting signals and pins of the opposite assertion.

If you already have NOT symbols on your schematic, you do not have to remove them. They will be ignored and the two signals on either side of the NOT symbol will be synonymed together.

Note: An unnamed net cannot be used with a NOT symbol unless you specify the size of the net with a SLASH symbol.

SIM_DIRECTIVES

Do not use the SIM_DIRECTIVES symbol. It is an old symbol that was used to pass simulator directives to RapidSIM.

Other Symbols

For information about	See
ALIAS	Creating an alias for a signal
IOPORT, INPORT, LNKPORT, BUFPORT, OUTPORT, AOUTPORT	Adding Ports
MERGE	Merge Symbols
SLASH	Specifying the Size of Nets
SLICE	Signal Slices (Bit and Part Selects)

Customizing Standard Library Symbols

To customize a Standard library symbol,

- Copy the symbol to a new cell and edit it. You can change the shape of the symbol, but do not change its visible or invisible properties.

  OR

- Create a new symbol and copy the properties from the corresponding Standard library symbol to the new symbol.

4

Error Checking Features in Design Entry HDL

Overview of the Error Checking Feature

Design Entry HDL performs the following error checking functions when generating the netlist for a design:

- Cross-View Checking on page 173
- Entity Declaration Checking for Instantiated Components on page 175

You can specify additional error-checking options in the *Check* tab of the *Design Entry HDL Options* dialog box.

To specify additional error checking options in Design Entry HDL

1. Choose *Tools – Options*.

   The *Design Entry HDL Options* dialog box appears.

2. Select the *Check* tab and specify the error checking options.

   For more information on error-checking options, see the help for the Design Entry HDL Options—Check dialog box.

Cross-View Checking

Cross-view checking is an error-checking feature of ports, port modes, and port types that is performed between the schematic views and the symbol views.

When you save a schematic, Design Entry HDL compares the ports, port modes, and port types in the schematic views and the symbol views and checks if they are consistent. If there are any inconsistencies, Design Entry HDL displays an error or warning message.

- Design Entry HDL displays the following error message if all the ports present in the schematic views are not present in the symbol views:

  ```
  168 ERROR Schematic has port but port does not exist in the symbol. Either
  delete this port from the schematic or add this port in the symbol.
  ```

  To correct this error, do one of the following:

  - ❑ Delete extra ports in the schematic views
  - ❑ Add missing ports in the symbol views
  - ❑ Check the Low Assertion Character settings in Part Developer

    Consider the following example:

    The symbol and entity of a cell have _N in the low (negative) assertion pin names. Part Developer has the following Low Assertion Character settings (Tools – Setup):

    - ❍ Additional Read = _N
    - ❍ Read/Write = *

    Before any modifications in Part Developer, the low asserted pin names have _N. This is in sync with the signal names on the schematic pages of the block. When the symbol is modified in Part Developer, the `_N` in the low asserted pins is changed to `*` in the symbol and the entity. This results in a mismatch in the schematic and the entity.

    In case you get this error and the Low Assertion Character settings are as described, perform the following to correct the problem:

  a. Set the following Low Assertion Character settings in Part Developer **before** editing the symbol in Part Developer:

    - ❍ Additional Read = _N
    - ❍ Read/Write = _N

  b. Make the required changes to the symbol.

  c. Save the symbol.

  d. Exit Part Developer.

  e. Export the design from Design Entry HDL.

- Design Entry HDL displays the following warning message if all the ports present in the symbol views are not present in the schematic views:

  ```
  171 WARNING Port exists in symbol but not in the schematic. Either delete this
  port from the symbol or add this port in the schematic.
  ```

To correct this error, do one of the following:

- ❑ Add missing ports in the schematic views
- ❑ Delete extra ports in the symbol views

If you have less ports in the schematic views than in the symbol views, Design Entry HDL displays a warning message because the additional ports found on the symbol are left open in the netlist.

Note: In the *Markers* dialog box, if you click on the warning message that is displayed when ports that are present in the symbol views are not present in the schematic views, the additional ports on the symbol will not get highlighted in Design Entry HDL. However, you can view the port name and port type of the additional ports in the *Markers Detail* dialog box.

- Design Entry HDL displays the following error message if the port mode declared in the schematic view is different from the port mode declared in the symbol view:

```
169 ERROR Port mode for the pin on the symbol is different from that on the pin
of the instance. Modify the port mode to make it the same.
```

  To correct this error, modify the port mode of the symbol or the schematic view to make it the same.

- Design Entry HDL displays the following error message if the port type declared in the schematic view is different from the port type declared in the symbol view:

```
170 ERROR Port type specified in the schematic and symbol is different. Modify
schematic/symbol to make port type same.
```

  To correct this error, modify the port type of the symbol or the schematic view to make it the same.

Entity Declaration Checking for Instantiated Components

When Design Entry HDL generates the VHDL netlist for a schematic, it also generates a VHDL component declaration for each component used in the schematic. To generate this component declaration, Design Entry HDL reads the entity declaration associated with the component. For example, if the part `NAND2` from the `/usr/libs/lsttl` library is instantiated in a schematic, Design Entry HDL reads the following file:

```
/usr/libs/lsttl/nand2/entity/vhdl.vhd
```

When Design Entry HDL has to read the entity declaration, there could be different situations:

- If an entity declaration for the part is found, the declaration is used to construct a component declaration. If you add the `VHDL_GENERICS` property to an instance on the schematic and if the same property is not found in the entity or the symbol of the component, it is declared in the component declaration section of the component.

- If no entity declaration is found and the component is placed in the schematic and saved, the following warning message is displayed:

  ```
  177 WARNING: Entity declaration for part does not exist in library
  ```

  The VHDL architecture is created and the component declaration section in the VHDL architecture (`sch_n/vhdl.vhd`) in this case is constructed from the symbol view of the component (`sym_n/symbol.css`). Because the symbol view does not contain information about the port mode of the component, Design Entry HDL declares all the ports as `inout` in the component declaration section.

- If the entity declaration is not present for the instance and the property `VHDL_GENERICS` is attached to the instance, the VHDL architecture will declare these generics in the component section in VHDL architecture (`sch_n/vhdl.vhd`), which gets constructed from the symbol view of the component (`sym_n/symbol.css`). Because the symbol view does not contain information about the port mode of the component, Design Entry HDL declares all the ports as `inout` in the component declaration section.

Netlisting Errors

This section describes some select netlisting errors that appear when you netlist the design for simulation. Each error is listed in the ascending order of the error numbers. The solution or workaround for the error is described.

Click on an error message for detailed information on the error message.

Error
ERROR 90: Error in parsing body file. Error in reading <path_to_symbol>symbol.css.
ERROR 105: SUPPLY_1 and SUPPLY_0 signal names must begin with a / (forward slash).
ERROR 106: SUPPLY_1 and SUPPLY_0 signals cannot be vectored.
ERROR 110: Range direction needs to be changed to ‘TO’.
ERROR 111: Range direction needs to be changed to ’DOWNTO’.
VHDL_ERROR 112: Value of VHDL_PORT property is set improperly.
ERROR 113: Port is attached to an unnamed signal.
ERROR 114: Port is not connected.

Error
VHDL_ERROR 118: Signal has two different VHDL_VECTOR_TYPE values.
VHDL_ERROR 119: Signal has two different VHDL_SCALAR_TYPE values.
ERROR 120: Signal is declared to be both a port and an alias.
WARNING 121: Port has two different port modes. Port being declared as inout.
WARNING 122: Signal is a global signal in one place but not in the other place. The signal will be treated as a global signal at both the places.
ERROR 123: Same alias is made to two different signals.
ERROR 124: Signal is declared to be both a scalar and a vector.
ERROR 126: Identifier is used as both a PATH value and a signal name.
WARNING 127: Identifier is used as both a component name and a signal name.
ERROR 129: A global signal cannot also be a port.
ERROR 131: You cannot tap from an unnamed signal.
ERROR 132: You cannot tap from an unconnected signal.
ERROR 136: The signal coming out of the concatenation symbol must be unnamed.
ERROR 137: Each pin on a concatenation symbol must be connected to a signal.
ERROR 144: Alias symbol has an unconnected pin.
ERROR 145: Pin on alias symbol has an unnamed signal attached.
ERROR 146: Signal coming out of an alias symbol is also a port.
FERROR 147: Signal coming out of an alias symbol cannot be a global signal.
ERROR 150: Signal connected to pin has incorrect width.
ERROR 151: Entity declaration for instance declares a port that is not on the instance.
ERROR 152: Port on instance does not exist in entity declaration for instance.
ERROR 153: Port on instance is vectored but port in entity declaration for instance is not.
ERROR 154: Port on instance is scalar but port in entity declaration for instance is not.
ERROR 155: Range direction for port on instance conflicts with port in entity declaration for instance.
ERROR 156: Instance port and entity port modes are incompatible.

Error
ERROR 158: Sizeable pin cannot be represented in Verilog because it is partly unconnected.
ERROR 164: Pin width is greater than attached signal width.
ERROR 165: Concatenated signal width must match pin width.
ERROR 166: Attached signal width is not an integer multiple of pin width.
ENTITY ERROR 169: Port mode for the pin on the symbol is different from that on the pin of the instance. Modify the port mode to make it the same.
ERROR 174: Output of tap is unconnected.
WARNING 177: Entity declaration for part does not exist in library.
ERROR 178: Port exists in the entity but not on the instantiated symbol. Please rewrite the necessary pages.
ERROR 179: Two signal names are attached to this net.
ERROR 181: Signals on both the sides of the MERGE/TIE symbol are connected to driver pins of other instances.
ERROR 182: Signal on one side of the MERGE/TIE symbol is a global signal and signal on the other side is connected to driver pin of another instance.
ERROR 183: Both sides of the MERGE/TIE symbol are connected to global signals.
WARNING 184: A OUTPORT symbol in the schematic must not be connected to MERGE/TIE symbol.
ERROR 185: One of the pins of MERGE/TIE symbol is unconnected.
ERROR 187: Signals attached to MERGE/TIE symbol have parameterized width.
ERROR 188: Signals attached to each side of MERGE/TIE symbol have different width.
WARNING 190: Signals on both the sides of the MERGE/TIE symbol are undriven.
WARNING 191: Cannot place pin properties on a pin with parameterized width if other pins on the instance have the same basename.
WARNING 192: Cannot place properties on specific bits of a signal which has parameterized width.
ERROR 197: Property on declarations symbol has incorrect value.
ERROR 198: Signal syntax is incorrect.
ERROR 205:Tap off bus is out of range. The HDL file being generated is incorrect.

Error
ERROR 206: The property on the DEFINE body must have an integer value.
ERROR 207: Property X_STEP not present on DEFINE instance.
ERROR 208: Property SIZE not present on DEFINE instance.
WARNING 211: Size Property not present on instance. Assuming a value of 1.
ERROR 212: Net connected to the output of Replicate instance should be unnamed or have the same width as the Replicate instance.
WARNING 217: Bit property not present on instance. Assuming a value of 0
ERROR 222: Error in symbol files.
ERROR 230: Net widths on both sides of the merge body do not match.
ERROR 231: Symbol pin is wider than the entity port.
ERROR 234: Different component uses same SPLIT_INST_NAME/SPLIT_INST prop value. Use different prop value for different components.
ERROR 260: Two assertion character - and * used in the signal name, it is not allowed. Use only one assertion character
ERROR 264: Property (SIZE/HAS_FIXED_SIZE/TIMES) can have only integer value. Ignoring this value and using 1 as the default value.
ENTITY_ERROR 267: Port range specified in the schematic and symbol is different. Modify schematic/symbol to make port range same.
ENTITY_ERROR 268: Port is specified vectored in the schematic but scalar on symbol. Modify schematic/symbol to make port consistent.
ENTITY_ERROR 269: Port is specified scalar in the schematic but vectored on symbol. Modify schematic/symbol to make port consistent.
ERROR 274: Instance name does not match the module name in the entity declaration.
ERROR 275: Two global signals are shorted.
WARNING 401: Binding Instance
ERROR 422: Chips File Packaging Error
ERROR 521: In Specifying Property On Instance
ERROR 526: In Specifying Split Inst Property on Instance

ERROR 90: Error in parsing body file. Error in reading <path_to_symbol>symbol.css.

Description

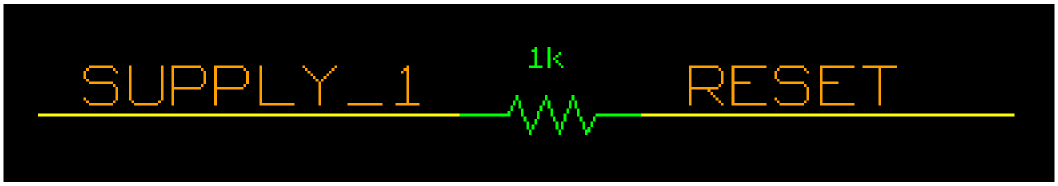
This error occurs when you run CRefer on a schematic which contains a block that does not have any instances on it.

ERROR 105: SUPPLY_1 and SUPPLY_0 signal names must begin with a / (forward slash).

Description

This error has occurred because of the following reasons:

- The SIG_NAME property on the pin of the symbol for the component has the value `SUPPLY_1` instead of `/SUPPLY_1`
- The SIG_NAME property on the pin of the symbol for the component has the value `SUPPLY_0` instead of `/SUPPLY_0`
- The signal you want to be declared as a SUPPLY0 net is named `SUPPLY_0` instead of `/SUPPLY_0`
- The signal you want to be declared as a SUPPLY1 net is named `SUPPLY_1` instead of `/SUPPLY_1`



In the above figure, the signal you want to be declared as a SUPPLY1 net is named `SUPPLY_1` instead of `/SUPPLY_1`. This resulted in the error.

Solution

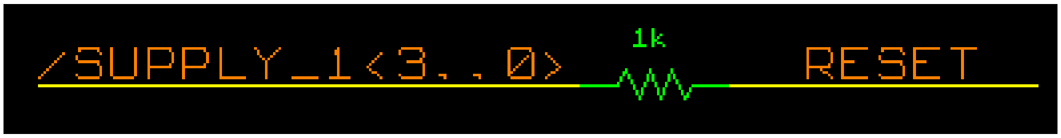
Ensure that the SUPPLY_1 and SUPPLY_0 signal names are declared as global signals, that is, begin with a forward slash (`/`) or have a `\G` suffix.

For more information on declaring SUPPLY0 nets, see SUPPLY_0 on page 166. For more information on declaring SUPPLY1 nets, see SUPPLY_1 on page 167.

ERROR 106: SUPPLY_1 and SUPPLY_0 signals cannot be vectored.

This error has occurred because of the following reasons:

- The SIG_NAME property on the pin of the symbol for the component has a vectored value, say `/SUPPLY1<3..0>`
- The SIG_NAME property on the pin of the symbol for the component has a vectored value, say `/SUPPLY0<3..0>`
- The signal you want to be declared as a SUPPLY0 net is vectored, say `/SUPPLY0<3..0>`
- The signal you want to be declared as a SUPPLY1 net is vectored, say `/SUPPLY1<3..0>`



In the above figure, the signal you want to be declared as a SUPPLY 1 net is a vectored signal. This resulted in the error.

Solution

Ensure that the /SUPPLY_1 and /SUPPLY_0 signal names are not vectored.

For more information on declaring SUPPLY 0 nets, see SUPPLY_0 on page 166. For more information on declaring SUPPLY 1 nets, see SUPPLY_1 on page 167.

ERROR 110: Range direction needs to be changed to ‘TO’.

Description

The range direction specified for the signal, port, or alias is not correct. For example, if the signal name is `DATA<0 DOWNTO 10>`, it is an error because DOWNTO should be used only to indicate a descending range direction.

Solution

Correct the range direction to TO. In the above example, change the signal name to:

```
DATA<0 TO 10>
```

ERROR 111: Range direction needs to be changed to ’DOWNTO’.

Description

The range specified to declare the width of the vectored port, signal or alias is not correct. You have specified a descending range but have given the range direction as ascending.

For example, if you specify the range for a vectored signal as `DATA<7 TO 0>`, this error is displayed because the syntax `TO` in the range specified indicates that it an ascending range, although it is a descending range.

Solution

Use the syntax `DOWNTO` to specify the range direction as descending. Taking the above example, the correct way to specify the descending range is `DATA<7 DOWNTO 0>`.

VHDL_ERROR 112: Value of VHDL_PORT property is set improperly.

Description

The value specified for the VHDL_PORT property is not correct.

Solution

Ensure that the value of the VHDL_PORT property is set to one of the following:

- IN
- AOUT
- BUFFER
- OUT
- INOUT
- LINKAGE

ERROR 113: Port is attached to an unnamed signal.

Description

The port is attached to an unnamed signal.

Solution

You should name the signal that is attached to a port. To do this:

➤ Name the signal with the *Wire – Signal Name* command.

ERROR 114: Port is not connected.

Description

The signal to which the port symbol is attached is not connected to any instance on the schematic.

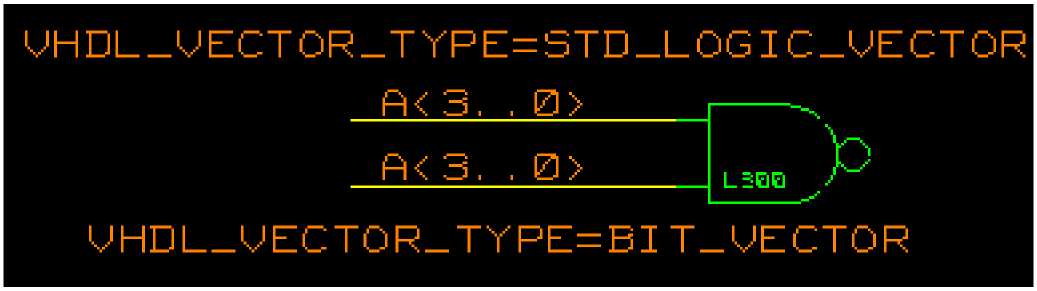
Solution

➤ Connect the signal to an instance on the schematic.

VHDL_ERROR 118: Signal has two different VHDL_VECTOR_TYPE values.

Description

Two different values are specified for the VHDL_VECTOR_TYPE property present on the same signal.


VHDL_VECTOR_TYPE=STD_LOGIC_VECTOR
A<3..0>
A<3..0>
L300
VHDL_VECTOR_TYPE=BIT_VECTOR

In the above figure, the value assigned to the VHDL_VECTOR_TYPE property on signal `A<3..0>` attached to the first pin of the LS00 is `STD_LOGIC_VECTOR`. However, the value assigned to the VHDL_VECTOR_TYPE property on the same signal `A<3..0>` attached to the second pin of the LS00 is `BIT_VECTOR`. This resulted in the error.

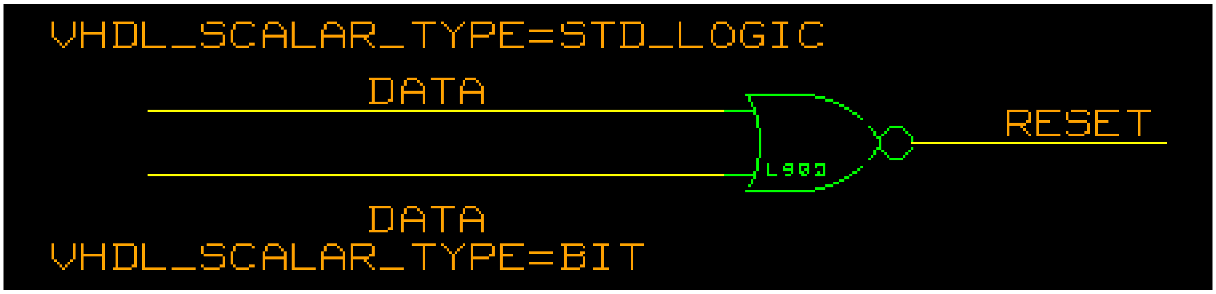
Solution

Ensure that the same value is specified for the VHDL_VECTOR_TYPE property present on the same signal.

VHDL_ERROR 119: Signal has two different VHDL_SCALAR_TYPE values.

Description

Two different values are specified for the VHDL_SCALAR_TYPE property present on the same signal.



In the above figure, the value assigned to the VHDL_SCALAR_TYPE property on signal `DATA` attached to the first pin of the LS02 is `STD_LOGIC`. However, the value assigned to the VHDL_SCALAR_TYPE property on the same signal `DATA` attached to the second pin of the LS02 is `BIT`. This resulted in the error.

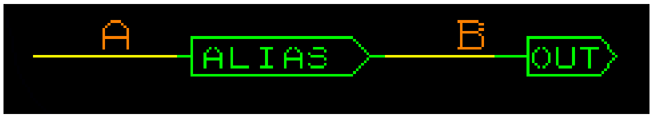
Solution

Ensure that the same value is specified for the VHDL_SCALAR_TYPE property present on the same signals.

ERROR 120: Signal is declared to be both a port and an alias.

Description

The signal connected to an ALIAS symbol is connected to a port.



In the above figure, signal B that is connected to an ALIAS symbol is connected to an OUTPORT symbol, resulting in an error.

Solution

You should not connect a signal that is connected to an `ALIAS` symbol to a port.

WARNING 121: Port has two different port modes. Port being declared as inout.

Description

Two different port modes have been declared for the port. This resulted in the port being declared as INOUT.

Solution

Check the properties on the pins of the port to ensure that only one port mode is declared for the port. To to this:

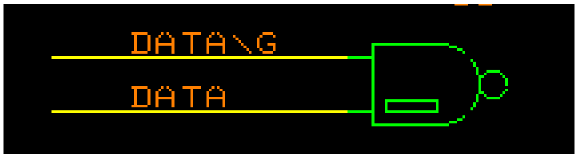
1. Choose *Text – Attributes*.
2. Click on a pin of the port to display the Attributes dialog box.
3. Verify if the VLOG_MODE or VHDL_MODE property has been used to declare more than one port mode.
4. Perform steps 2 and 3 above for each pin of the port.

   Note: If the port has several pins, the VLOG_MODE or VHDL_MODE property has to be attached to only one of the pins of the port.

WARNING 122: Signal is a global signal in one place but not in the other place. The signal will be treated as a global signal at both the places.

Description

The signal that is declared as a global signal (for example, `DATA\G` or `/DATA` is a global signal) in one place in the schematic is not declared as a global signal in another place in the schematic (for example, the signal is declared as `DATA` and not `DATA\G` in the other place). The signal will be treated as a global signal at both the places.



In the above figure, signal `DATA` has the same base name (DATA) as the global signal `DATA\G` resulting in the error.

Solution

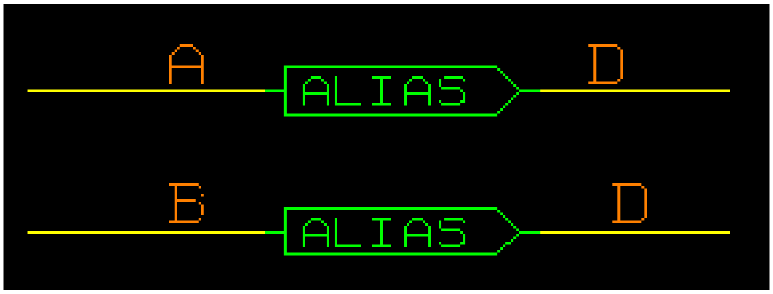
Change the signal names to ensure that no signal has the same name as a global signal.

Note: This error message will not be displayed if a signal connected to an instance that has the `HDL_POWER` property has the same name as the value of the `HDL_POWER` property. This is because, Design Entry HDL treats the signal as a global signal.

ERROR 123: Same alias is made to two different signals.

Description

The ALIAS symbol in the Standard library is used to specify another name for a signal. If the same alias is made to two different signals, this error occurs.



In the above figure, signal A is aliased to signal D and signal B is also aliased to signal D. In this case, signal A is driving signal D and signal B is also driving signal D. This results in the error.

Solution

Ensure that the same alias is not made to two different signals.

ERROR 124: Signal is declared to be both a scalar and a vector.

Description

The signal has been declared to be both a scalar and a vectored signal. This error is displayed if, for example, a schematic has two signals `DATA` (a scalar signal) and `DATA <7..0>` (a vectored signal).

Solution

In a schematic you cannot have a signal with the same name declared as a scalar and a vectored signal. Change the signal names in the schematic to ensure that no signal with the same name is not declared as scalar and vector.

ERROR 126: Identifier is used as both a PATH value and a signal name.

Description

The name of the signal is the same as `page<page_number>_<value of PATH property on any instance>`.

When Design Entry HDL generates the netlist for the design, it writes each component instance in the netlist as: `page<page_number>_<value of PATH property on the instance>`. For example, if the value of the PATH property of an instance on page 1 of the schematic is `i1`, the instance is written in the netlist as `page1_i1`. If the *Check InstanceVs Signal* check box in the *Verilog Netlist* or *VHDL Netlist* dialog box is selected, Design Entry HDL displays this error message for every signal that has the same name as `page<page_number>_<value of PATH property on any instance>`.

Solution

Change the signal name to ensure that it does not have the same name as `page<page_number>_<value of PATH property on any instance>`.

WARNING 127: Identifier is used as both a component name and a signal name.

Description

The signal name is the same as the name of a component used in the schematic page.

For example, if you have used the `ls04` component in the schematic page and also have a signal named `ls04` in the schematic page, this error occurs.

Solution

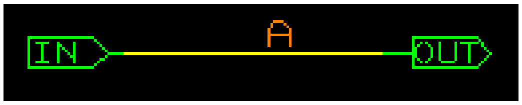
Change the signal name to ensure that it is not the same as the name of any component used in the schematic page.

ERROR 128: Net has two port symbols connected to it.

Description

The signal directly connects one port symbol to another port symbol.

If a signal is connected to a port, Design Entry HDL names the port using the signal name. If a signal directly connects two port symbols, Design Entry HDL will not be able to determine the type of the port.



In the above figure, signal A directly connects an INPORT symbol to an OUTPORT symbol, resulting in an error.

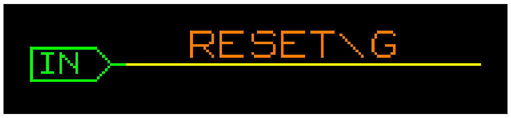
Solution

Ensure that the signal does not directly connect two port symbols.

ERROR 129: A global signal cannot also be a port.

Description

The signal connected to the port is a global signal.



In the above figure, signal `RESET\G` is connected to an INPORT symbol, resulting in an error.

Solution

You must not connect a global signal to a port symbol.

ERROR 131: You cannot tap from an unnamed signal.

Description

The signal to which the TAP symbol is attached is unnamed. You should name the signal that is attached to a TAP symbol.

Solution

➤ Name the signal with the *Wire – Signal Name* command.

For more information on using TAP symbols in your schematic, see TAP Symbols on page 154.

ERROR 132: You cannot tap from an unconnected signal.

Description

The signal to which the TAP symbol is attached is not connected to any instance on the schematic.

Solution

➤ Connect the signal to an instance on the schematic.

For more information on using TAP symbols in your schematic, see TAP Symbols on page 154.

ERROR 136: The signal coming out of the concatenation symbol must be unnamed.

Description

The signal coming out of the CONCAT symbol must be unnamed.

Solution

1. Choose *Edit – Delete*.
2. Click on the signal name to delete it.

For more information on using CONCAT symbols in your schematic, see CONCAT Symbols on page 160.

ERROR 137: Each pin on a concatenation symbol must be connected to a signal.

Description

One or more pins on a CONCAT symbol are not connected to a signal.

Solution

Ensure that all the pins on the CONCAT symbol are connected to a signal.

For more information on using CONCAT symbols in your schematic, see CONCAT Symbols on page 160.

To suppress this error, set the value of the `SUPPRESS_CONCAT_ERROR` directive to ‘`ON`’ in the `START_NETLIST...END_NETLIST` section of the `.cpm` file.

ERROR 144: Alias symbol has an unconnected pin.

Description

One of the pins of the ALIAS symbol is not connected to any signal.

Solution

Ensure that both the pins of the ALIAS symbol are connected to signals.

ERROR 145: Pin on alias symbol has an unnamed signal attached.

Description

An unnamed signal is attached to the pin of the ALIAS symbol. You should name the signal that is attached to the pin.

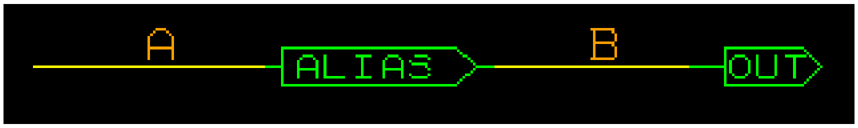
Solution

➤ Name the signal with the *Wire – Signal Name* command.

ERROR 146: Signal coming out of an alias symbol is also a port.

Description

The signal coming out of the ALIAS symbol is connected to a port symbol.



In the above figure, signal B is connected to an OUTPORT signal, resulting in an error.

Solution

You should not connect the signal coming out of an ALIAS symbol directly to a port symbol.

FERROR 147: Signal coming out of an alias symbol cannot be a global signal.

Description

The signal coming out of the ALIAS symbol is a global signal.

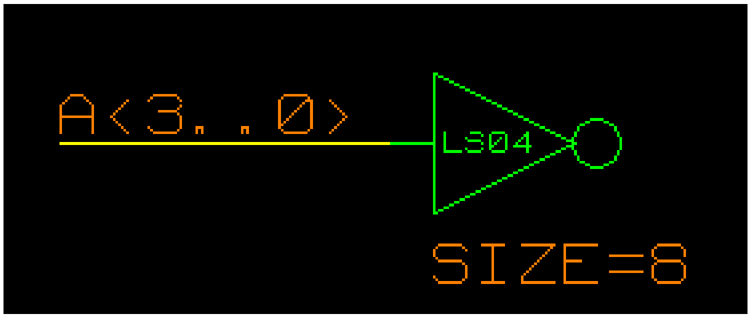
Solution

Ensure that the signal coming out of the ALIAS symbol is not a global signal.

ERROR 150: Signal connected to pin has incorrect width.

Description

The value of the SIZE property specified on the component determines the width of the pins on the component. If the width of the signal and the width of the pin of the component to which the signal is connected are not the same, this error occurs.



In the above figure, the value of the SIZE property on the component is 8. However, since the width of signal A is 4, this error occurs.

Solution

Ensure that the value of the SIZE property specified on the component is the same as the width of the signal.

ERROR 151: Entity declaration for instance declares a port that is not on the instance.

Description

The entity declaration for the instance (`/entity/verilog.v` or `/entity/vhdl.vhd` file) declares a port that is not present on the instance.

Solution

Open the symbol for the component in Design Entry HDL or Part Developer and add the port on the symbol, if required. Then save the symbol. This will bring your symbol and entity ports in sync.

ERROR 152: Port on instance does not exist in entity declaration for instance.

Description

The port present on the instance does not exist in the entity declaration for the instance (`/entity/verilog.v` or `/entity/vhdl.vhd` file).

Solution

Open the symbol for the component in Design Entry HDL or Part Developer and delete the port on the symbol, if it is not required. Then save the symbol. This will bring your symbol and entity ports in sync.

ERROR 153: Port on instance is vectored but port in entity declaration for instance is not.

Description

The port on the instance is a vectored port but the same port in the entity declaration for the instance (`/entity/verilog.v` or `/entity/vhdl.vhd` file) is not a vectored port.

Solution

Open the symbol for the component in Design Entry HDL or Part Developer and correct the port on the symbol of the instance. Then save the symbol. This will bring your symbol and entity ports in sync.

ERROR 154: Port on instance is scalar but port in entity declaration for instance is not.

Description

The port declared on the symbol instance is a scalar port but the same port in the entity declaration for the instance (`/entity/verilog.v` or `/entity/vhdl.vhd`) is not scalar.

Solution

Open the symbol for the component in Design Entry HDL or Part Developer and correct the port on the symbol of the instance. Then save the symbol. This will bring your symbol and entity ports in sync.

ERROR 155: Range direction for port on instance conflicts with port in entity declaration for instance.

Description

The range direction for the port on the instance conflicts with the range direction for the port in the entity declaration for the instance (the `verilog.v` or `vhdl.vhd` file in the `entity` view of the instance).

For example, if the range on the port on the instance is `<0 to 3>` and the range for the port in the entity declaration for the instance is `<3 downto 0>`, this error occurs.

Solution

Open the symbol for the component in Design Entry HDL or Part Developer and save the symbol. This will bring your symbol and entity ports in sync.

ERROR 156: Instance port and entity port modes are incompatible.

Description

There is a mismatch between the port mode defined in the component declaration for the instance and the entity declaration associated with the component.

The following port modes can be declared using the VHDL_MODE property:

- IN
- OUT
- BUF
- INOUT

Solution

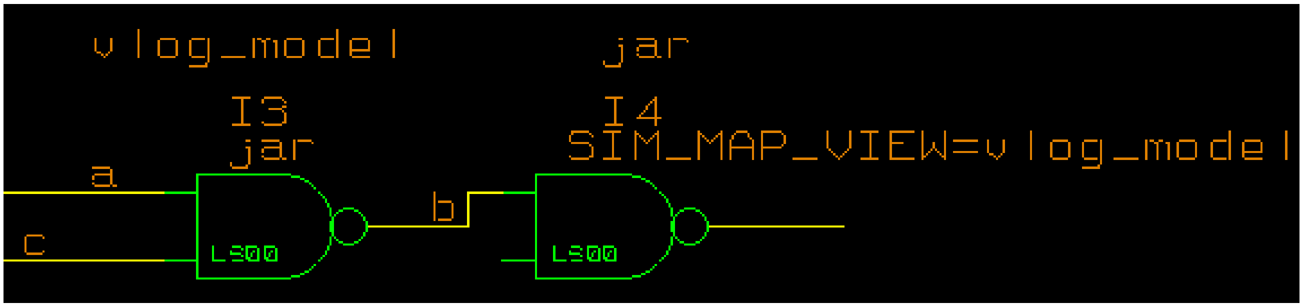
Verify that the following mismatches do not exist in the port mode declared in the instance and the entity:

- PORT declared as IN in entity and OUT, INOUT, or BUF on instance
- PORT declared as OUT in entity and IN, INOUT, or BUF on instance
- PORT declared as BUF in entity and IN, OUT, or INOUT on instance
- PORT declared as INOUT in entity and BUF on instance

ERROR 158: Sizeable pin cannot be represented in Verilog because it is partly unconnected.

Description

This error is generated when you use wrappers to simulate a sizeable part that has unconnected pins, and has either `SPLIT_INST_NAME` or `SPLIT_INST` property attached to it. The error is generated because Design Entry HDL netlister is unable to determine the width of open pins.



Solution

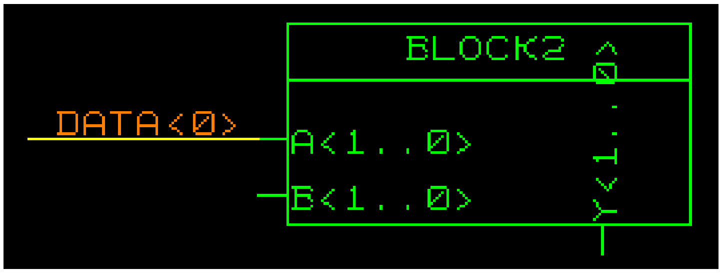
To remove the error, perform one of the following steps:

- Do not have unconnected pins in the schematic.
- Use `SPLIT_INST` and `SPLIT_INST_NAME` properties only on asymmetrical parts.

ERROR 164: Pin width is greater than attached signal width.

Description

The width of the signal attached to the pin is lesser than the width of the pin.



In the above figure, the width of pin `A<1..0>` is greater than the width of the signal `DATA<0>` attached to it, resulting in an error.

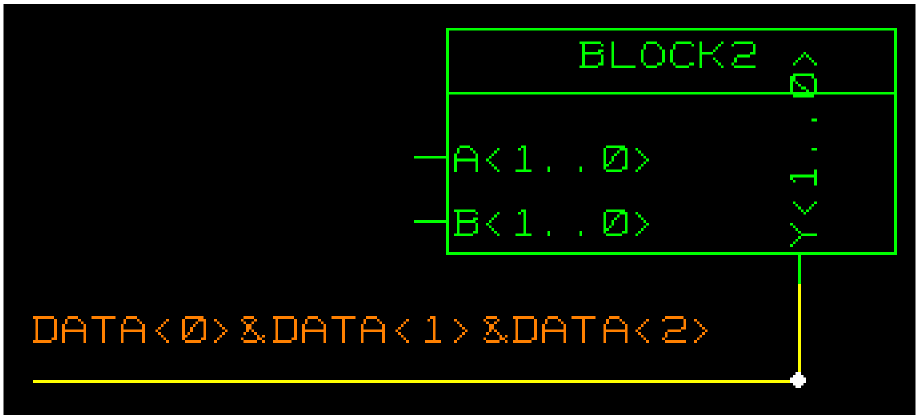
Solution

Ensure that the width of the signal attached to the pin is the same or an integral multiple of the width of the pin.

ERROR 165: Concatenated signal width must match pin width.

Description

The width of the concatenated signal attached to the pin is not the same as the width of the pin.



In the above figure, the width of pin `Y<1..0>` is lesser than the width of the concatenated signal `DATA<0>&DATA<1>&DATA<2>` attached to it, resulting in an error.

Solution

Ensure that the width of the concatenated signal attached to the pin is the same as the width of the pin.

In the above example, the width of the concatenated signal should be changed to `DATA<0>&DATA<1>`.

ERROR 166: Attached signal width is not an integer multiple of pin width.

Description

The width of the vectored signal is not an integer multiple of the width of a sizable pin to which it is attached.

For example, suppose that a vectored signal `S` is attached to pin `A` of component LS00. The property SIZE=4 is attached to pin `A`. In this case, the pin width is 1, but if the signal width is not 4 (not an integer multiple of the pin width 4 x 1), this error occurs.

Solution

Change the width of the signal to be an integer multiple of the width of the sizable pin.

ENTITY ERROR 169: Port mode for the pin on the symbol is different from that on the pin of the instance. Modify the port mode to make it the same.

Description

The port mode for the pin on the symbol and the corresponding pin on the instance are different.

Suppose that you have created a symbol from a schematic or a schematic from a symbol. This error message will appear if, later on, you do one of the following:

- Change the port mode for a pin on the symbol but do not make the same change in the port mode of the corresponding pin on the instance.

- Change the port mode for a pin on the instance but do not make the same change in the port mode of the corresponding pin on the symbol.

For example, if the port mode specified on the pin on the symbol is `VHDL_MODE=IN` and the port mode for the corresponding pin on the instance is `VHDL_MODE=OUT`, this error occurs.

Solution

The port mode for a pin on the symbol must always be the same as the port mode of the corresponding pin on the instance.

- If the pin on the symbol has the correct port mode, do one of the following:
  - ❑ Choose *Tools – Generate View* to regenerate the schematic from the symbol.
  - ❑ Open the schematic in Design Entry HDL, correct the port mode on the pin on the instance to make it the same as the port mode of the corresponding pin on the symbol and save the schematic.
- If the pin on the instance has the correct port mode, do one of the following:
  - ❑ Choose *Tools – Generate View* to regenerate the symbol from the schematic.
  - ❑ Open the symbol in Design Entry HDL, correct port mode on the pin on the symbol to make it the same as the port mode of the corresponding pin on the instance and save the symbol.

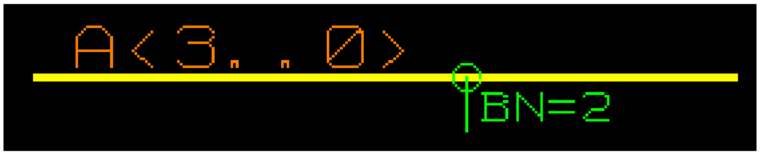
The allowed values for the `VHDL_MODE` property on a pin are `IN`, `OUT`, `INOUT`, `LINKAGE`, and `BUF`. The allowed values for the `VLOG_MODE` property on a pin are `INPUT`, `OUTPUT`, `INOUT,LINKAGE`, and `BUF`.

If you want to specify the port mode for an interface signal on a schematic, attach one of the symbols `INPORT`, `OUTPORT`, `IOPORT` or `BUFPORT` from the Standard library to the interface signal.

ERROR 174: Output of tap is unconnected.

Description

The output pin of the tap symbol (CTAP, BIT TAP, LSB TAP or MSB TAP symbols) is not connected to any signal.



In the above figure, the output pin of the CTAP symbol is not connected to any signal, resulting in an error.

Solution

Connect the output pin of the tap symbol to a signal.

For more information on tap symbols, see TAP Symbols on page 154.

WARNING 177: Entity declaration for part does not exist in library.

Description

When Design Entry HDL generates the VHDL netlist for a schematic Design Entry HDL cross checks the entity and symbol views. it also generates a component declaration for each component used in the schematic. To generate this component declaration, Design Entry HDL reads the entity declaration associated with the component. For example, if the part `NAND2` from the `/usr/libs/lsttl` library is instantiated in a schematic, Design Entry HDL reads the `/usr/libs/lsttl/nand2/entity/vhdl.vhd` file to generate the VHDL component declaration.

If the entity declaration file is not found this warning message is displayed.

The VHDL netlist still gets created and the component declaration section in the VHDL netlist (`/sch_1/vhdl.vhd`) is constructed from the symbol view of the component (`/sym_1/symbol.css`). Since the symbol view does not contain information about the port mode of the component, Design Entry HDL declares all the ports as `inout` in the component declaration section of the netlist.

Solution

Open the symbol for the component in Design Entry HDL or Part Developer and save it. The entity declaration file (`/entity/vhdl.vhd`) for the component are created in the entity view of the component.

ERROR 178: Port exists in the entity but not on the instantiated symbol. Please rewrite the necessary pages.

Description

The entity declaration for the instance (`/entity/verilog.v` or `/entity/vhdl.vhd` file) declares a port that is not present on the symbol for the instance.

Solution

Open the symbol for the component in Design Entry HDL or Part Developer and add the port on the symbol, if required. Then save the symbol. This will bring your symbol and entity ports in sync.

To save the symbol in Design Entry HDL, do the following:

1. In Design Entry HDL, choose *File – Open*.

   The *View Open* dialog box appears.
2. Select the library in which the component exists in the *Library* drop-down.

   The list of components in the library are displayed.
3. Select the component for which the error occurred.

   The component name is displayed in the *Cell* field.
4. Select *Symbol* from the *View* drop-down.
5. Click *Open* to view the symbol in Design Entry HDL.
6. Choose *File – Save*.

ERROR 179: Two signal names are attached to this net.

Description

Two signal names are attached to the wire.

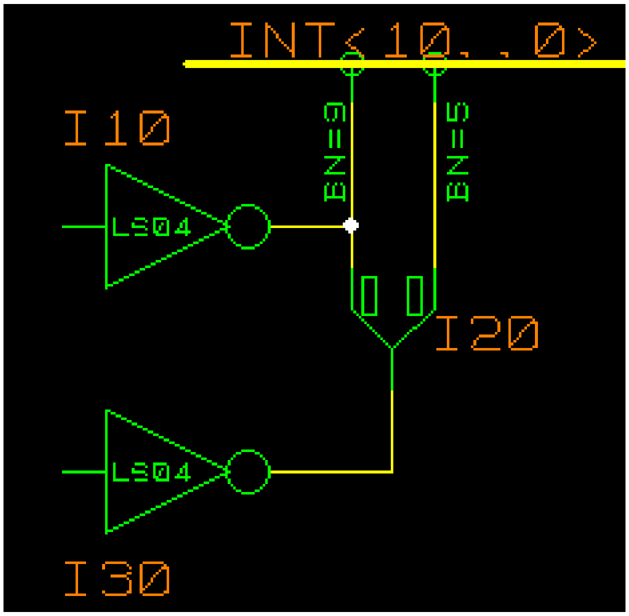
Solution

Remove one of the signal names attached to the wire.

ERROR 181: Signals on both the sides of the MERGE/TIE symbol are connected to driver pins of other instances.

Description

The signals on both the sides of the MERGE or TIE symbol are connected to driver (output) pins of other instances.



In the above figure, the signal connected to the input pin of the 2 MERGE symbol is connected to the driver (output) pin of instance `I10` of the `ls04` component. The signal connected to the output pin of the MERGE symbol is also connected to the driver pin of instance `I30` of the `ls04` component. This error occurs because the signals on both the sides of the MERGE symbol are connected to driver pins of instances of the `lso4` component.

Solution

If the signals on one side of the MERGE or TIE symbol are connected to driver pins of other instances, ensure that the signals on the other side of the MERGE or TIE symbol are not connected to driver pins of other instances.

ERROR 182: Signal on one side of the MERGE/TIE symbol is a global signal and signal on the other side is connected to driver pin of another instance.

Description

The signal on one side of the MERGE or TIE symbol is a global signal and the signal on the other side of the MERGE or TIE symbol is connected to the driver pin of another instance.

Solution

If the signal on one side of the MERGE or TIE symbol is a global signal, ensure that the signal on the other side of the MERGE or TIE symbol is not connected to the driver pin of other instances.

ERROR 183: Both sides of the MERGE/TIE symbol are connected to global signals.

Description

Both the sides of the MERGE or TIE symbol is connected to global signals.

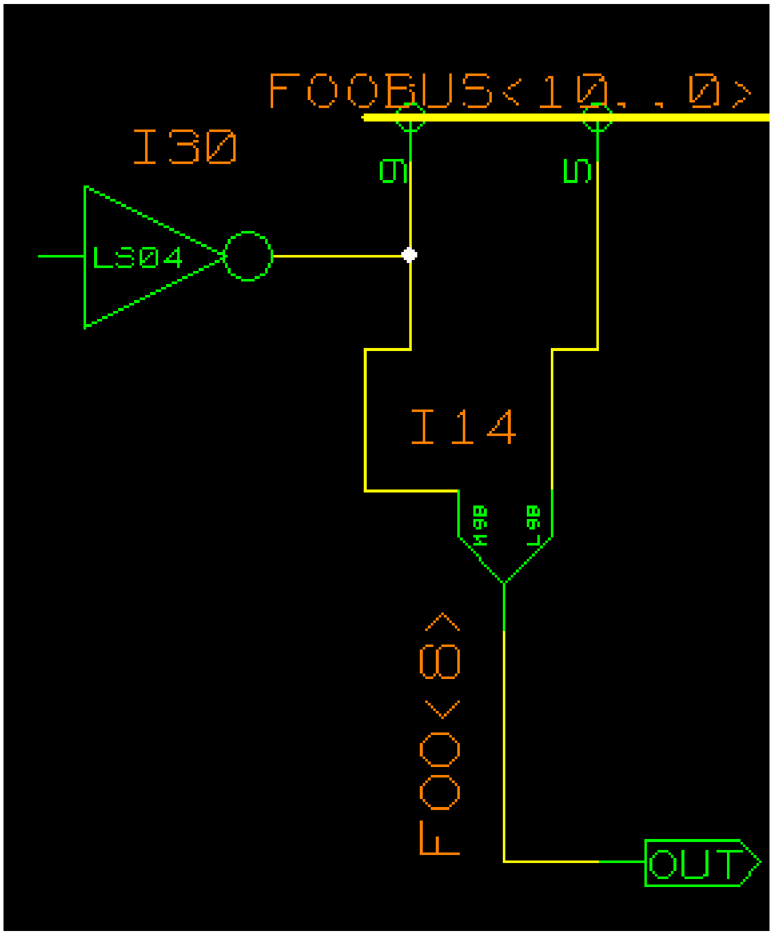
Solution

Ensure that both the sides of the MERGE or TIE symbol are not connected to global signals.

WARNING 184: A OUTPORT symbol in the schematic must not be connected to MERGE/TIE symbol.

Description

This error occurs because the output of the MERGE or TIE symbol is directly connected to an OUTPORT symbol.



In the above figure, the output of the 2 MERGE symbol is connected to an OUTPORT symbol. This results in an error.

Solution

Do not connect the output of a MERGE or TIE symbol directly to an OUTPORT symbol.

ERROR 185: One of the pins of MERGE/TIE symbol is unconnected.

Description

One of the pins of the MERGE or TIE symbol is not connected to any signal.

Solution

Connect a signal to the pin of the MERGE or TIE symbol.

ERROR 187: Signals attached to MERGE/TIE symbol have parameterized width.

Description

The signal attached to the MERGE or TIE symbol has parametrized width. For example, if a parametrized signal `CLOCK<SIZE-1..0>` is attached to the MERGE or TIE symbol, this error occurs.

Solution

Ensure that the signals connected to the MERGE or TIE symbol do not have parametrized width.

ERROR 188: Signals attached to each side of MERGE/TIE symbol have different width.

Description

The width of the signals attached to each side of the MERGE or TIE symbol are not the same. For example, if the width of the signal attached to one side of the TIE symbol is `<3..0>` and the width of the signal attached to the other side of the TIE symbol is `<7..0>`, this error will occur.

The sum of the size of the signals connected to the input pins of a MERGE symbol must be equal to the size of the signal connected to the output pin of the MERGE symbol.

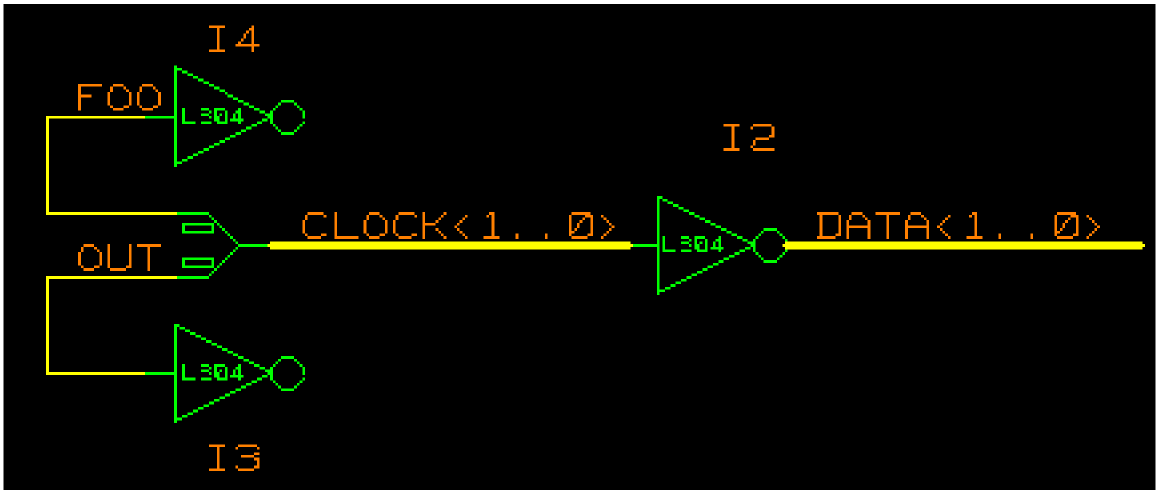
Solution

Ensure that the width of the signals attached to each side of the MERGE or TIE symbol are the same.

WARNING 190: Signals on both the sides of the MERGE/TIE symbol are undriven.

Description

The signals on both the sides of the MERGE/TIE symbol are undriven.



In the above figure, the signal `FOO` connected to the input pin of the 2 MERGE symbol is connected to the receiver (input) pin of instance `I4` of the `ls04` component. The signal `CLOCK<1..0>` connected to the output pin of the MERGE symbol is also connected to the receiver (input) pin of instance `I2` of the `ls04` component. This warning occurs because the signals on both the sides of the MERGE/TIE symbol are undriven.

Solution

If the signals on one side of the MERGE or TIE symbol are connected to receiver (input) pins of components, ensure that the signals on the other side of the MERGE or TIE symbol connected only to the driver (output) pins of components.

To rectify the error, you might want to change the direction of pins of a component. You can do this by editing the symbol of the component. When you edit the symbol and save it, the pin direction gets updated in the `verilog.v` and `vhdl.vhd` files in the entity view of the component.

WARNING 191: Cannot place pin properties on a pin with parameterized width if other pins on the instance have the same basename.

Description

If a property is attached to a vectored pin of an instance that has parametrized width and another pin of the instance has the same base name, this warning is displayed.

For example, suppose there are two pins `A<1..0>` and `A<2>` on a component. If a property is attached to pin `A<1..0>`, it is not clear if you want the same property on pin `A<2>` also. This results in the warning message being displayed.

Solution

Ensure that pins on a component do not have the same base name. Taking the above example, rename pin `A<2>` as `B<2>`.

WARNING 192: Cannot place properties on specific bits of a signal which has parameterized width.

Description

If a property is attached to a bit of a vectored signal that has parametrized width and another signal on the design has the same base name, this warning is displayed.

For example, suppose there are two signals `A<3..0>` and `A<2>` in a design. If a property is attached to signal `A<2>`, it is not clear if you want the same property on signal `A<3..0>` also. This results in the warning message being displayed.

Solution

Ensure that signals in your design do not have the same base name. Taking the above example, rename pin `A<2>` as `B<2>`.

ERROR 197: Property on declarations symbol has incorrect value.

Description

A property defined on the DECLARATIONS symbol has an incorrect value.

Solution

See the details of the error message in the *Markers* window to know the name of the property that has the incorrect value. Refer to the *PCB and IC Packaging Properties Reference* for information on the supported values for the property.

ERROR 198: Signal syntax is incorrect.

Description

There is a syntax error in the signal name. For example, if the signal name is `~DATA`, this error is displayed.

Solution

Ensure that the signal name complies with the conventions for naming of signals in Design Entry HDL.

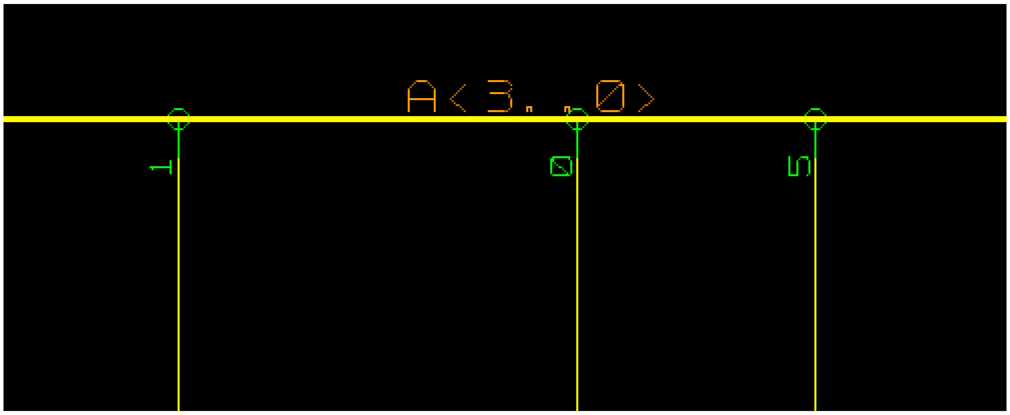
ERROR 205:Tap off bus is out of range. The HDL file being generated is incorrect.

Description

The bit number (BN) property attached to the pin of a TAP symbol has a value outside the range of the bus being tapped.

Example

In the following figure, the BN property attached to the pin is 5, whereas the bus range is between 0 and 3.



Solution

Change the value of the BN property such that it lies in the bus range. In the above example, the value 5 should be changed to a value between 0 and 3.

Other Possible Causes

You might have tapped off a bus, which is not interpreted as a bus. This could happen because of the specific syntax not being recognized as vector syntax.

Example

`DATA(7..0)` or `ADDR[15..0]`

Solution

Ensure that the `MULTI_FORMAT` directive is set to `'ON'` in the `.cpm` file. You no longer see the HDLDirect warnings on saving the schematic.

ERROR 206: The property on the DEFINE body must have an integer value.

Description

One of the two properties, X_STEP or X_FIRST, on a DEFINE body has a non-integer value.

Solution

Change the non-integer value of the property to an integer value.

ERROR 207: Property X_STEP not present on DEFINE instance.

Description

The default property X_STEP, is not present on the symbol of component DEFINE and hence, on an instance of the component.

Solution

Remove the instance of component DEFINE. Add the X_STEP property on the symbol of component DEFINE and instantiate it again.

ERROR 208: Property SIZE not present on DEFINE instance.

Description

The default property SIZE is not present on the symbol of component DEFINE and hence, on an instance of the component.

Solution

Remove the instance of component DEFINE. Add the SIZE property on the symbol of component DEFINE and instantiate it again.

WARNING 211: Size Property not present on instance. Assuming a value of 1.

Description

This error is generated when an instance has either HDL_REPLICATE = TRUE, HDL_LSBTAP = TRUE, or HDL_MSBTAP = TRUE property attached to it without the SIZE property.

The HDL_REPLICATE property, which is attached to the origin of a symbol, classifies the symbol as REPLICATE. The REPLICATE symbol is used while making models for sizeable parts. Therefore, the HDL_REPLICATE property must always be used along with the SIZE property.

Similarly, with HDL_LSBTAP and HDL_MSBTAP properties, the SIZE property is required to specify the width of the tapped signal.

Solution

If the SIZE property is not specified, by default the tool assumes the value to be 1. To change the size to a value other than 1:

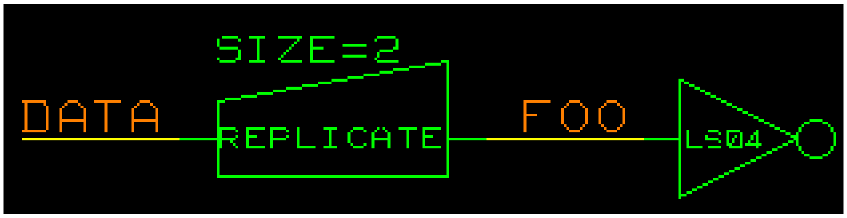
1. Select *Text – Attributes*.
2. Click on the instance to which the SIZE property is to be added.

   The *Attributes* dialog box appears.
3. Click *Add*.
4. Enter the *Property Name* as SIZE and the *Value* as 2.
5. Click OK.

ERROR 212: Net connected to the output of Replicate instance should be unnamed or have the same width as the Replicate instance.

Description

The net connected to the output of the REPLICATE symbol is named or does not have the same width as the REPLICATE symbol.



In the above figure, the REPLICATE symbol has the width 2 (denoted by the SIZE=2 property). This error occurs because the signal `FOO` connected to the output pin of the REPLICATE symbol does not have the same width as the REPLICATE symbol.

Solution

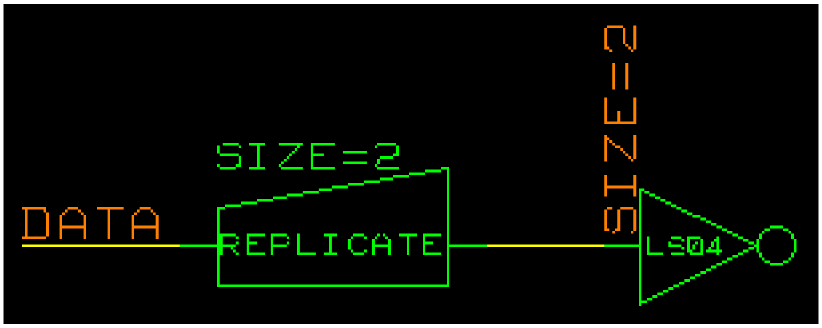
Do one of the following:

- Ensure that the signal connected to the output pin of the REPLICATE symbol has the same width as the REPLICATE symbol.

  In the above example, change the name of the signal to `FOO<1..0>`.

- Do not name the signal connected to the output pin of the REPLICATE symbol.

  If you do not name the signal connected to the output pin, ensure that the pin of the instance to which the output signal is connected has the same width as the REPLICATE symbol. In the above example, if the output signal is not named, add the `SIZE=2` property on the input pin of the `ls04` component.



WARNING 217: Bit property not present on instance. Assuming a value of 0

Description

If the value of the HDL_BITTAP property specified on an instance is TRUE, Design Entry HDL reads the value of the BIT property specified on the instance to determine the bit to be tapped from the bus connected to the instance. If the BIT property is not present on the instance, Design Entry HDL assumes that 0 bits have to be tapped from the bus and displays this warning message.

Solution

Add the BIT property on the instance to specify the bit to be tapped from the bus connected to the instance.

ERROR 222: Error in symbol files.

Description

An error occurred when reading the `symbol.css` file in the symbol view of the component.

Solution

Open the symbol in Design Entry HDL and save it to regenerate the symbol files. You can also open the symbol in the Part Developer tool and save it to regenerate the symbol files. For more information, see the Part Developer User Guide.

To save the symbol in Design Entry HDL, do the following:

1. In Design Entry HDL, choose *File – Open*.

   The *View Open* dialog box appears.
2. Select the library in which the component exists in the *Library* drop-down.

   The list of components in the library are displayed.
3. Select the component for which the error occurred.

   The component name is displayed in the *Cell* field.
4. Select *Symbol* from the *View* drop-down.
5. Click *Open* to view the symbol in Design Entry HDL.
6. Choose *File – Save*.

ERROR 230: Net widths on both sides of the merge body do not match.

Description

The combined width of the signals connected to the input pins of the MERGE symbol is not the same as the width of signal connected to the output pin of the MERGE symbol.

Solution

Correct the width of the signal connected to the output pin of the MERGE symbol.

ERROR 231: Symbol pin is wider than the entity port.

Description

The width of the pin on the symbol is wider than the width of the port in the entity. For example, if the width of the pin on the symbol is `a<3..0>` and the width of the port specified in the `verilog.v` or `vhdl.vhd` file in the entity view of the part is `a<2..0>`, this error occurs.

Solution

Open the symbol in Design Entry HDL and correct the width of the pin on the symbol. You can also open the symbol in the Part Developer tool and correct the width of the pin on the symbol. For more information, see the Part Developer User Guide.

To correct the width of the pin on the symbol in Design Entry HDL, do the following:

1. In Design Entry HDL, choose *File – Open*.

   The *View Open* dialog box appears.

2. Select the library in which the component exists in the *Library* drop-down.

   The list of components in the library are displayed.

3. Select the component for which the error occurred.

   The component name is displayed in the *Cell* field.

4. Select *Symbol* from the *View* drop-down.
5. Click *Open* to view the symbol in Design Entry HDL.
6. Correct the width of the pin of the symbol.
7. Choose *File – Save*.

ERROR 234: Different component uses same SPLIT_INST_NAME/SPLIT_INST prop value. Use different prop value for different components.

Description

The SPLIT_INST_NAME property value for two different components are the same.

Solution

The SPLIT_INST_NAME property is used on all split components of a large pin count device that have to be merged into a single instance in the netlist. The value of the SPLIT_INST_NAME property must be the same on such split components. This error occurs if the value of the SPLIT_INST_NAME property on split components of two different large pin count devices is the same.

For example, suppose there are two large pin count devices `ASYM_PART` and `ASYM_PART1`. The device `ASYM_PART` is split into two components-`INST1` and `INST2`. The device `ASYM_PART1` is split into three components-`INST_A`, `INST_B`, and `INST_C`. If the same value is specified for the SPLIT_INST_NAME property on component `INST1` of device ASYM_PART and component `INST_A` of device ASYM_PART1, this error occurs.

ERROR 260: Two assertion character - and * used in the signal name, it is not allowed. Use only one assertion character

Description

You can use the `*` character as a suffix or the `-` character as a prefix in a signal or a pin name to declare a low-asserted signal or pin. If both the assertion characters (`-` and `*`) are used in the signal or pin name, this error occurs.

Solution

Use only one of the assertion characters in the signal or pin name.

Note: You can also use a `_N` suffix in a signal or a pin name to declare a low-asserted signal or pin. Cadence recommends that you use a `_N` suffix to indicate a low-asserted signal or pin.

ERROR 264: Property (SIZE/HAS_FIXED_SIZE/TIMES) can have only integer value. Ignoring this value and using 1 as the default value.

Description

The SIZE, HAS_FIXED_SIZE, and TIMES properties can have only integer values. Although you can assign alphanumeric values to these properties, only those alphanumeric values where the first letter is a numeral are allowed. In such cases, only the numeric part of the value is considered and rest is ignored.

Property Value	**Design Entry HDL netlister......**
SIZE = 3A	Sets the size to 3.
Size = 3A3	Sets the size to 3.
SIZE = A3	Generates an error.

Solution

Assign numeric values to SIZE, HAS_FIXED_SIZE, and TIMES properties.

ENTITY_ERROR 267: Port range specified in the schematic and symbol is different. Modify schematic/symbol to make port range same.

Description

The range of a port on the symbol is different from the range of the corresponding port on the schematic.

Suppose that you have created a symbol from a schematic or a schematic from a symbol. This error message will appear if, later on, you do one of the following:

- Change the range of a port on the symbol but do not make the same change in the range of the corresponding port on the schematic.
- Change the range of a port on the schematic but do not make the same change in the range of the corresponding port on the symbol.

For example, if the port range specified on the symbol is `ADDRESS<3..0>` and the port range on the schematic is `ADDRESS<7..0>`, this error occurs.

Solution

The range of a port on the symbol must always be the same as the range of the corresponding port on the schematic.

- If the symbol has the correct port range, do one of the following:
    - ❑ Choose *Tools – Generate View* to regenerate the schematic from the symbol.

- ❑ Open the schematic in Design Entry HDL, correct the port range on the schematic to make it the same as the port range on the symbol and save the schematic.

- If the schematic has the correct port range, do one of the following:
  - ❑ Choose *Tools – Generate View* to regenerate the symbol from the schematic.
  - ❑ Open the symbol in Design Entry HDL, correct the port range on the symbol to make it the same as the port range on the schematic and save the symbol.

ENTITY_ERROR 268: Port is specified vectored in the schematic but scalar on symbol. Modify schematic/symbol to make port consistent.

The port on the schematic is a vectored port, but the corresponding port on the symbol is a scalar port.

Suppose that you have created a symbol from a schematic or a schematic from a symbol. This error message will appear if, later on, you do one of the following:

- Change a vectored port to a scalar port on the symbol, but do not make the same change for the corresponding port on the schematic.
- Change a scalar port to a vectored port on the schematic, but do not make the same change for the corresponding port on the symbol.

For example, if the port on the schematic is `ADDRESS<7..0>` (a vectored port) and the port on the symbol is `ADDRESS` (a scalar port), this error occurs.

Solution

The ports on the schematic and the symbol must always be in sync.

- If you want to declare the port as scalar, do one of the following:
  - ❑ Choose *Tools – Generate View* to regenerate the schematic from the symbol.
  - ❑ Open the schematic in Design Entry HDL, change the vectored port on the schematic to make it a scalar port and save the schematic.
- If you want to declare the port as vectored, do one of the following:
  - ❑ Choose *Tools – Generate View* to regenerate the symbol from the schematic.
  - ❑ Open the symbol in Design Entry HDL, change the scalar port on the symbol to make it a vectored port and save the symbol.

ENTITY_ERROR 269: Port is specified scalar in the schematic but vectored on symbol. Modify schematic/symbol to make port consistent.

Description

The port on the schematic is a scalar port, but the port on the symbol is a vectored port.

Suppose that you have created a symbol from a schematic or a schematic from a symbol. This error message will appear if, later on, you do one of the following:

- Change a vectored port to a scalar port on the schematic, but do not make the same change for the corresponding port on the symbol.
- Change a scalar port to a vectored port on the symbol, but do not make the same change for the corresponding port on the schematic.

For example, if the port on the schematic is `ADDRESS` (a scalar port) and the port on the symbol is `ADDRESS<7..0>` (a vectored port), this error occurs.

Solution

The ports on the schematic and the symbol must always be in sync.

- If you want to declare the port as scalar, do one of the following:
  - ❑ Choose *Tools – Generate View* to regenerate the symbol from the schematic.
  - ❑ Open the symbol in Design Entry HDL, change the vectored port on the symbol to make it a scalar port and save the symbol.
- If you want to declare the port as vectored, do one of the following:
  - ❑ Choose *Tools – Generate View* to regenerate the schematic from the symbol.
  - ❑ Open the schematic in Design Entry HDL, change the scalar port on the schematic to make it a vectored port and save the schematic.

ERROR 274: Instance name does not match the module name in the entity declaration.

Description

The cell name does not match with the module name in the entity view of the cell.

This can happen if you rename the cell; the module name in the entity view remains unchanged and thus out of synchronization with the cell name.

Solution

Open the symbol for the cell and save it. Saving the symbol rewrites the entity view and updates the module name in the entity with the new cell name.

ERROR 275: Two global signals are shorted.

Description

The global signal is shorted with another global signal.

Solution

Two global signals should not be shorted. If you want to short the global signals, add them in the *Allowed Global Shorts* list of the *Design Entry HDL Options* dialog box. Design Entry HDL will not display this error message if the global signals listed in the *Allowed Global Shorts* list are shorted.

This error message will also be displayed when you netlist the design for digital simulation, if you have specified two shorted global signals `CLOCK\G` and `SWITCH\G` in the *Allowed Global Shorts* list and have also added `CLOCK\G` in the *Supply 0* list and `SWITCH\G` in the *Supply 1* list in the *Verilog* netlisting options dialog box.

WARNING 401: Binding Instance

The view specified by the `SIM_MAP_VIEW` ***property for instance <instance_name>, pathval: <path_value> does not exist or doesn't have a valid map file. The specified property, <property_name and value> is being ignored.***

Description

This is a warning message that is generated in the following two scenarios:

- ❑ There is no view corresponding to the view name specified by the value assigned to the `SIM_MAP_VIEW` property.
- ❑ The view exists but does not have a valid `verilog.map` file.

In both the above mentioned cases the default configuration is used for binding the component instances.

Solution

To remove this error message you must specify a view name that has a valid map file, or remove the `SIM_MAP_VIEW` property. To avoid incorrect simulation results, it is recommended that these warnings must be removed before simulating the design.

ERROR 422: Chips File Packaging Error

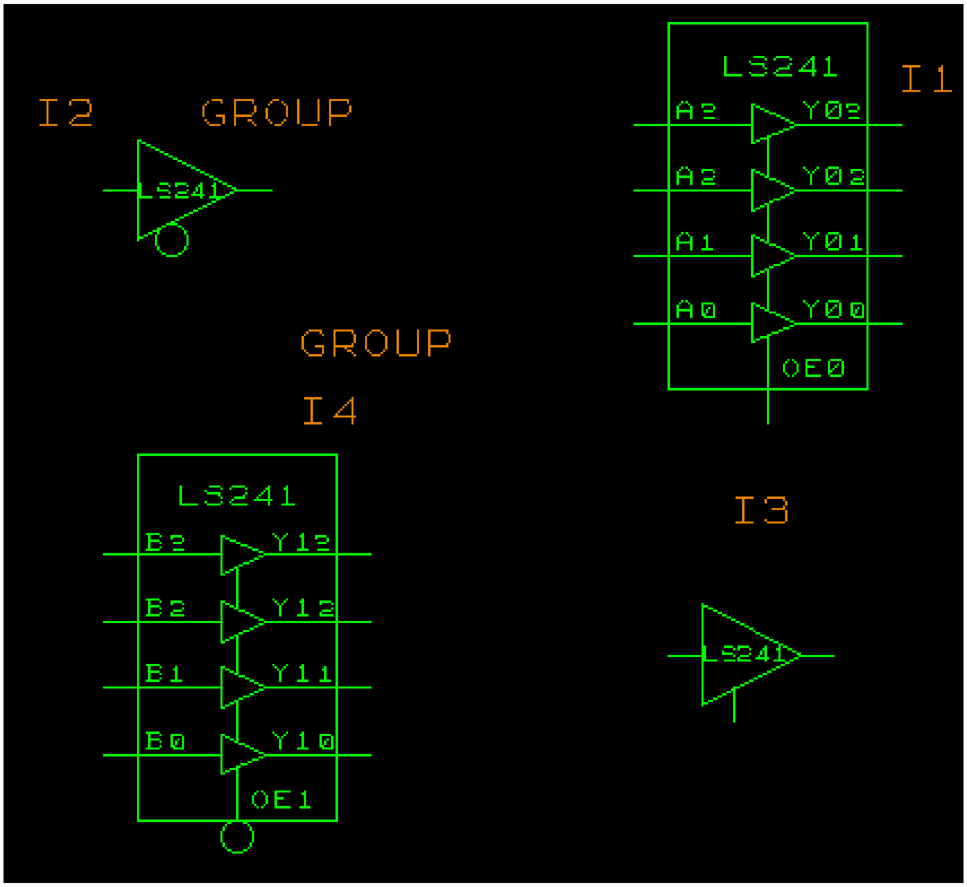
Cannot package primitive instance: <instance_name>, pathval: <path_value>. Ignoring split inst property.

Description

This error is generated when `SPLIT_INST` or `SPLIT_INST_NAME` properties are used in a manner that violates the package description in the `chips.prt` file.

For example, according to the `chips.prt` file of `LS241`, from among the parts shown below, I2 and I4 can only be grouped with either I1 or I3. If I2 and I4 are grouped together, Chips File

Packaging Error is generated because you have more than one entry for a pin name which violates the description in the `chips.prt` file.



Solution

Before grouping parts in the same split inst group, read the `chips.prt` file to find out the parts that can be grouped together and then use the `SPLIT_INST` or `SPLIT_INST_NAME` property to group two or more parts in the same split inst group.

ERROR 521: In Specifying Property On Instance

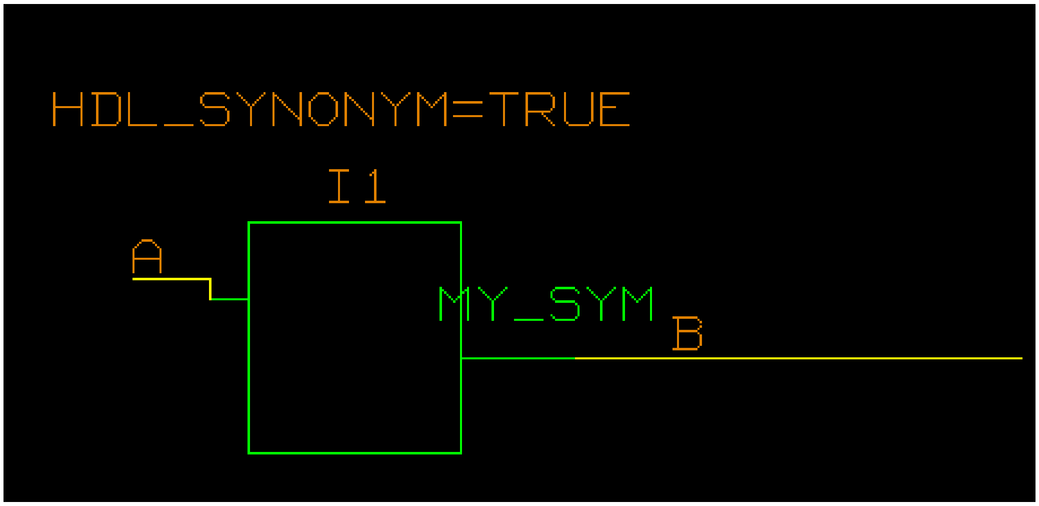
This message appears when ever the properties specified on an instance are not correct. This message may appear either as an error or a warning. Depending on the situation in which the message appears, the message has different explanations. Some of the explanations are listed below:

Incorrect SYNONYM property on instance <component_name>. The SYNONYM property can be specified only on components with exactly 2 pins. The HDL_SYNONYM property is being ignored.

Description

This warning is generated when the HDL_SYNONYM property is attached to a symbol that has more than one input or output pins. This is because the symbols or the parts with HDL_SYNONYM = TRUE are used to specify a different name for the same signal.

A symbol with correct usage of HDL_SYNONYM = TRUE and the corresponding Verilog netlist are shown below:



Verilog Netlist:

```
wire a;
wire  b;


  wire  page1_a;
  wire  page1_b;

  assign page1_a = a;
  assign page1_b = b;

  assign a   = b;
```

Solution

Add HDL_SYNONYM property only to components and symbols that have exactly one input and one output pin. To know more about SYNONYM, see *Using the Standard Library Symbols* in *Allegro Design Entry HDL User Guide*.

Incorrect property specified for instance <instance_name>, pathval <path_value>. The specified property REMOVE = LINK is being ignored. The REMOVE property can only be used with components having exactly one input and one output pin.

Description

This error is generated when REMOVE = LINK is attached to a component that does not satisfy the criterion of one input and one output pin. The REMOVE property is attached only to the two pin components. Resistor packs with one input and multiple outputs are the only exception to this rule. You can add REMOVE = LINK to such resistor packs without error.

Solution

Remove the REMOVE property from the component. To know more about the REMOVE property, see *Simulation Properties* section of *PCB and IC Packaging Properties Reference*.

Incorrect property specified for instance: <instance_name>, pathval <path_value>. The specified property REMOVE = AUTO is being ignored. The REMOVE property can only be used with parts having exactly one input and one output pin.

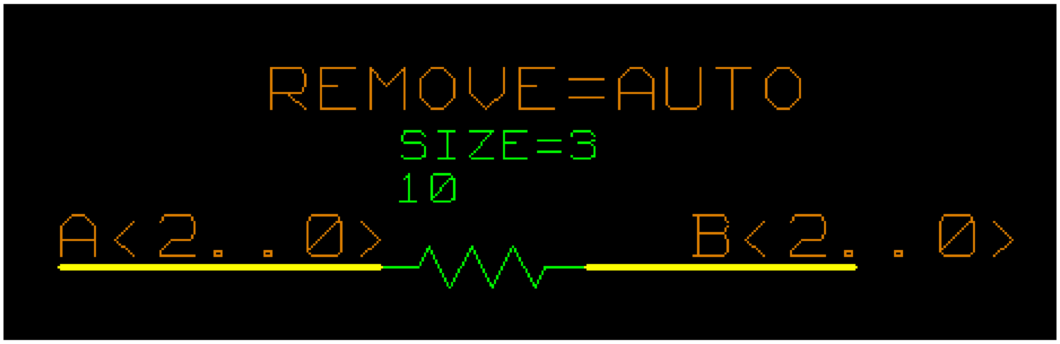
Description

This error is generated when REMOVE = AUTO is attached to a component that does not satisfy the criterion of one input and one output pin. The REMOVE property can be attached only to the two pin components.

Resistor packs with one input and multiple outputs are the only exception to this rule. You can add REMOVE = AUTO to such resistor packs without error.

Note: In most of the resistor packs, REMOVE = AUTO property is assigned at the Origin and cannot be removed.

The REMOVE property also works fine if the size property is used on a two pin part. A part of the schematic that has REMOVE= AUTO attached to a resistor with SIZE = 3 is shown below:



Solution

Delete the REMOVE = AUTO property attached to the part with more than one input or out pin.

Incorrect property value specified for instance <instance_name>, pathval <path_value>. The specified property is being ignored. Supported property values pairs are REMOVE = LINK, REMOVE = AUTO, REMOVE = FALSE, and REMOVE = EXCLUDE.

Description

This warning is generated when the REMOVE property is assigned a value other than LINK, AUTO, EXCLUDE, or FALSE. In this case, Design Entry HDL netlister ignores the REMOVE property while creating the netlist.

Solution

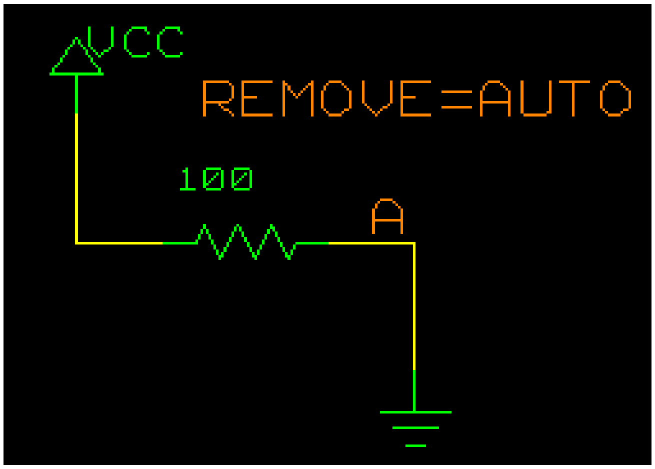
Change the value assigned to the REMOVE property to either LINK, AUTO, or EXCLUDE. To know more about the REMOVE property, see Simulation Properties section of *PCB and IC Packaging Properties Reference*.

Power net connections specified for instance: <instance_name>, pathval: <path_value>. One side of the instance is connected to supply 1 and the other side to supply0. The REMOVE= AUTO property can only be used with parts having one input

and one output each of that are connected to different nets. Remove one of the power net connections for this instance from the schematic.

Description

This error is generated when REMOVE = AUTO is attached to a part that is connected to a power source on one side and ground on the other side. Adding REMOVE = AUTO on the resistors shorts the connection between the power supply and ground because of which the error is generated.



Solution

To remove the above error you can do any one of the following:

1. Remove the ground element and ensure that the attached wire has a signal name.
2. Remove the Power supply and ensure that the attached wire has a signal name.

Unable to use PORT_ORDER property for instance: <instance_name>, pathval <path_value>. The directive MAP_BY_POSITION is being ignored.

Description

This warning is generated when you select the *Position Mapping* check box in the *Netlist* tab, and the PORT_ORDER information is not available in the map file corresponding to a part used in the schematic. In this case, the netlist is created using names instead of positions.

Solution

To prevent occurrence of this warning, ensure that the part is bibded to a view that has the PORT_ORDER information in the map file. To remove the warning, you can:

- modify the map file to add the PORT_ORDER information.
- use SIM_BND_VIEW or SIM_MAP_VIEW property to bind the component to a view containing map file with the PORT_ORDER information.
- clear the *Position Mapping* check box.

Note: To know more about the PORT_ORDER property see *Simulation Properties* in *Allegro Design Entry HDL Digital Simulation User Guide*.

Incorrect property specified for instance: <instance_name>, pathval <path_value>. The property PORT_ORDER is being ignored.

Description

This error is generated when the PORT_ORDER is available in the map file but none of the pins, in the PIN_MAP section of the map file, match to the ports listed in the PORT_ORDER. This error is generated only when the pinlist is NULL. The error will not be generated even if there is a single pin that is mapped to a port listed in the PORT_ORDER list. In the netlist, only those pins that are mapped to some port listed in the PORT_ORDER list appear, rest of the pins are ignored by the Design Entry HDL netlister.

A part of the map file for a component is shown below:

```
FILE_TYPE=VERILOG_MAP;
...
...
PROPERTY
              PORT_ORDER = ’(I1,I2,O1)’;
              COMPONENT=’SN74LS00’;
              RANGE;
         END_PROPERTY;
         PIN_MAP
       ’B’<0>=’(u2)’;
       ’A’<0>=’(u1)’;
       ’-Y’<0>=’(r1)’;
     END_PIN;
     END_MODEL;
```

```
END_PRIMITIVE;
END.
```

In the `verilog.map` file shown above, ports listed in the PORT_ORDER section are I1, I2, and O1. Ports to which the pins are mapping are u2, u1, and r1. As there is no common port, this will generate error.

Solution

Modify the map file to either change the PORT_ORDER information or the PIN_MAP information.

Incorrect property specified for instance: <instance_name>, pathval: <path_value>.The property HDL_REPLICATE is being ignored.

The property HDL_REPLICATE can be added only on instances that have two pins. The input pin must be scalar with pin name INPUT. The output pin must be sizeable like PINNAME<SIZE-1..0>. The SIZE property on the instance will determine the number of times the signal connected to pin INPUT is to be replicated.

Description

This error is generated when you add HDL_REPLICATE = TRUE on a part that either has:

- ❑ more that one input or output pins.
- ❑ input pin is not scaler.
- ❑ output pin is not sizeable.

Solution

Remove the HDL_REPLICATE = TRUE property from the part that does not satisfy the above mentioned criterion. To know more about the HDL_REPLICATE property, see *PCB and IC Packaging Properties Reference*.

ERROR 526: In Specifying Split Inst Property on Instance

LOCATION not specified for instance. The specified SPLIT_INST property is being ignored. Specify SPLIT_INST and LOCATION property on the instance

Description

The use model for the SPILT_INST property is that it has to be used with the LOCATION property that specifies the hard location. The value of the location property is then used as the split inst group name.

Solution: Along with SPLIT_INST = TRUE, add the LOCATION property with the same value to all the components that need to form a split inst group. To know more about working with the SPLIT_INST property, see *Working with SPLIT_INST and SPLIT_INST_NAME properties* in *Allegro Design Entry HDL Simulation User Guide*.

Dual SPLIT property specified for instance. The specified property: SPLIT_INST = TRUE is being ignored. Specify either SPLIT_INST_NAME = <value> or SPLIT_INST = TRUE and LOCATION = <value>.

Description

This error is generated when an instance has both SPLIT_INST_NAME, and SPLIT_INST and LOCATION, properties attached to it. You cannot attach both these properties on a single instance. To know more about SPLIT_INST and SPLIT_INST_NAME properties, see *Working with SPLIT_INST and SPLIT_INST_NAME properties* in *Allegro Design Entry HDL Simulation User Guide*.

Solution

Remove either the SPLIT_INST_NAME property or the SPLIT_INST and LOCATION property from the instance.

5

Design Entry HDL Files

This chapter describes the syntax of files that are created or used by Design Entry HDL.

System Initialization File

The Cadence Project Manager (`.cpm`) file is used to initialize front-end tools that have several setup options as well as tools that require knowledge about the current project.

See the *Project Manager User Guide* for more information.

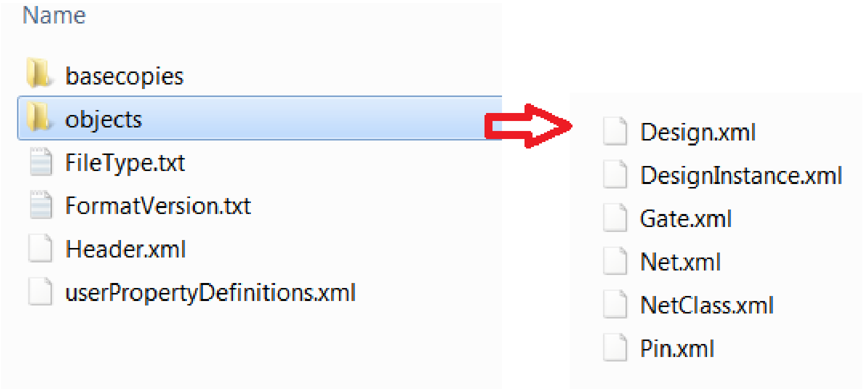
Cadence Library File

The `cds.lib` file contains a list of libraries to be used in conjunction with the project specified in the project file.

See the *Project Manager User Guide* for more information.

Constraint and Property Data File

The Design Constraint File, `.dcfx` (`.dcf`), contains information of all the constraints and properties applied to objects in a design. When you save the design, this file is created in the `sch_1` folder in the ZIP format. The `.dcfx` (`.dcf`) file is a binary file, which, when extracted, creates multiple XML and text files that contain constraints and properties information.



Name

basecopies
objects
FileType.txt
FormatVersion.txt
Header.xml
userPropertyDefinitions.xml

Design.xml
DesignInstance.xml
Gate.xml
Net.xml
NetClass.xml
Pin.xml

The base copy files that contain constraints information of each block in the design are also preserved inside the `basecopies` folder. These base copy files are used to compare constraints when running the back to front flow.

Viewing a DCF File

The `.dcf` file can be used to review constraints and properties applied to different objects in a design. By converting the binary `.dcf` file to an XML file, you can view all the information in a single place.

To convert the binary `.dcf` into a text file, run the following command from the system command prompt:

```
cmfeedback -debugView <file>.dcf
```

This command converts the binary `.dcf` file to an XML file with `.dcf.dcfx` extension, which you can view in any text editor. This file is only for viewing and cannot be replaced with the original `.dcf` file.

If you want the `.dcf.dcfx` file to automatically open in a text editor, you can set the path to a text editor using an environment variable, as follows:

```
set CM_DEBUG_EDITOR = C:\Program Files (x86)\Notepad\notepad.exe
```

ASCII Design Data Files

An ASCII design data file is one of the design database files that Design Entry HDL creates when you write a drawing. This file is read in if the design has no binary design data file (also referred to as a cell). ASCII design data files represent all drawings except symbol (SYM) drawings.

Design data files consist of commands to add to each object in a drawing. Design Entry HDL re-creates a drawing by reading the commands in the ASCII design data file. You can edit the file to modify your drawing.

The Design Entry HDL name for the ASCII file is SCH with version and page number extensions (for example, <*library*>.*cell*.SCH.1.1.csa).

ASCII design data files contain:

- File identification and end statements

  Each ASCII logic file starts with this line to identify the file type:

  ```
  FILE_TYPE = MACRO_DRAWING;
  ```

  The file ends with:

  ```
  QUIT
  ```

- Object definitions

Each type of object in a Design Entry HDL drawing has a specific definition format.

Note: If you generate ASCII files using a tool other than Design Entry HDL, discrepancies might creep in if the text size format used is different from the Design Entry HDL format. For example, the text size values might be stored as a decimal value, such as .2 or .5. Design Entry HDL stores text size as 47 multiplied with the text size in the ASCII file. If this number is not a whole number, it is rounded off.

For example, the `text size = .8` is internally stored in Design Entry HDL as `47 * .8 = 38`(rounded from 37.6). However, when you run the *hier_write* command, Design entry HDL reads the text size from the ASCII file and writes it as `38/47 = .8048` while generating the `.csb` or `.csv` files. All the rounded off text sizes are modified when you run the *hier_write* command.

ASCII Object Descriptions For Objects

Components

```
forceadd name
[R angle]
pt ;
[paint color pt]
```

Note: `forceadd` is used so that a placeholder is created if the component is not found. `forceadd` works only in ASCII drawing files and not in Design Entry HDL scripts.

name	The component name includes the version number.
R angle	Rotation of the added component. The *`angle`* definition is optional. `0, 90, 180, 270`
pt	Location of the component on the drawing.
`paint color pt`	(Optional) Included if the component color differs from the default.

Wires

`wire` *`linetype pattern pt1 pt2 ;`*

`linetype` This numeric argument includes the line color and thickness definition. If the number is converted to binary, the least significant bit is the thin/heavy bit (0 = thin, 1 = heavy). The remaining seven bits specify the color.

`pattern` Fill pattern of the line. The values this parameter can take and the resulting line patterns are shown in the figure below.

`-1`

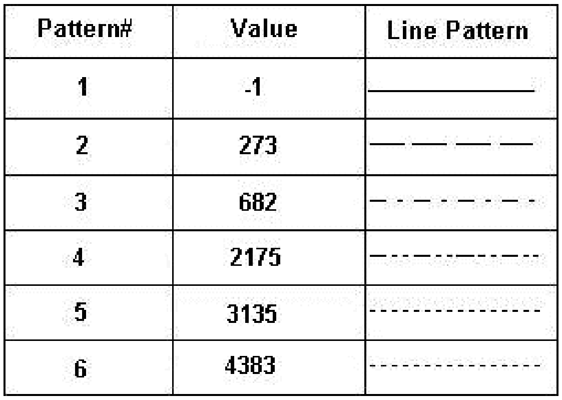
`273`

`682`

`2175`

`3135`

`pt1 pt2` The begin and end points of the wire.



Pattern#	Value	Line Pattern
1	-1	________________
2	273	—— —— —— ——
3	682	— - — - — - — -
4	2175	—·-·—·-·—·-·—·-·
5	3135	----------------
6	4383	----------------

Dots

```
dot type pt ;
[paint color pt]
```

`type` If type is 0, the dot is open; if type is 1, the dot is filled. If the type is neither 0 or 1, Design Entry HDL assumes the dot is open.

pt	Location of the dot on the drawing.
paint color pt	(Optional) Included if the dot color differs from the default.

Circles and Arcs

```
circle pt1 pt2 ; (for circles)
[paint color pt]
or
circle pt1 pt2 pt3 ; (for arcs)
[paint color pt]
```

pt	Location of the circle or arc on the drawing.
paint *color pt*	(Optional) Included if the color of the circle or arc differs from the default.

Notes

```
forcenote contents pt angle;
[display size pt ;]
[paint color pt]
```

Note: The `forcenote` command is similar to the note command in the Design Entry HDL editor except that the `forcenote` command terminates after reading one note. `forcenote` works only in ASCII drawing files and not in Design Entry HDL scripts.

contents	Note text.
pt	Location of the note on the drawing.
angle	Rotation of the added note.
	`0, 90, 180, 270`
display *size pt*	This line is included if the note is not the default size. This command makes the text the correct size.
paint *color pt*	(Optional) Included if the note color differs from the default.

Properties

```
forceprop default_status last name value
[R angle]
```

```
[J justification_type]
pt ;
[display size pt ;]
[paint color pt]
```

default_status — Handles changes to properties on library components and can be

0 User-defined property

1 Property is coming from the symbol

2 Property added by Design Entry HDL

`last` | `lastpin` | `lastprop` — Property is to be attached to the last object or wire entered.

- Argument `last` is added to a general property
- Argument `lastpin` followed by a *`pt`* that describes the location of the pin in absolute coordinates is added for a pin property
- Argument `lastprop` is added to a property that is owned by another property. An example of such a property is the `XR` property that is attached to the `SIG_NAME` property by CRefer.

R *`angle`* — Rotation of the added property. The *`angle`* definition is optional.

`0, 90, 180, 270`

J *`justification_type`* — Justification of the added text can be

0 Left-justified text

1 Centered text

2 Right-justified text

If not specified, the property is created with the current default justification. If an illegal value is given, the property is left justified.

`pt` — Location of the property on the drawing.

display size *`pt`* — Sets the visibility of the property name and value.

paint color *`pt`* — (Optional) Included if the property color differs from the default.

Bubbled Pins

The description of bubbled pins in the ASCII file is

```
forcebubble pt ...
```

All pins that are not in their default bubbled state are listed.

Note: forcebubble works only in ASCII drawing files and not in Design Entry HDL scripts.

Within ASCII design data files, Design Entry HDL internal coordinates are 0.00175 inches per unit. Points are represented by their coordinates, enclosed in parentheses and separated by a space. For example, the point x=100, y=200 becomes (100 200).

Angles are represented by a number from zero through seven on components only. For text objects, the angle values can be between zero through three only.

Angle Representation

0	0 degrees
1	90 degrees
2	Mirror of 0 degrees
3	Mirror of 270 degrees
4	180 degrees
5	270 degrees
6	Mirror of 180 degrees
7	Mirror of 90 degrees

Binary Design Data Files

These files contain the same information as the corresponding ASCII file but in a proprietary binary format that is quicker for Design Entry HDL to read and save. The Design Entry HDL name for the ASCII file is SCH with version and page number extensions (for example, <*library*>.*cell*.SCH.1. 1.csb).

Symbol File (symbol.css)

This section explains the syntax of the symbol file, saved on the system as `symbol.css`. The symbol file contains descriptions for following objects in ASCII format:

- Lines
- Arcs
- Text
- Connections
- Component Properties
- Pin Properties
- Bubble Groups

Design Entry HDL internal coordinates are 0.00175 inches per unit.

ASCII Symbol Descriptions For Objects

Lines

Lines require one line each in the symbol file. The format for a thin line is:

```
L x1 y1 x2 y2 [pattern] color
```

The format for a thick line is

```
M x1 y1 x2  y2 [pattern] color
```

where

x1 y1 x2 y2 The line’s endpoint coordinates; the line runs from *(x1 y1)* to *(x2 y2)*.

pattern

(Optional) Identifies the line style (solid, broken, and so on) as a bit pattern.

`-1`

`273`

`682`

`2175`

`3135`

color

Internal Design Entry HDL color number. The line type describes both the color and thickness of the line. When the integer is converted to a binary value, bit 0 defines the thickness (0 = thin), and the seven most significant bits define the color.

1 Black (default)

3 Red

5 Green

9 Blue

17 Yellow

33 Orange

67 Salmon

69 Violet

71 Brown

73 Skyblue

75 White

77 Peach

81 Pink

83 Purple

97 Aqua

99 Gray

Arcs

The format for an arc is:

```
A x y radius start_angle stop_angle color
```

where

x y radius — Center and radius points of the arc.

start_angle/ stop_angle — Floating point numbers that measure the angles in degrees counterclockwise from the x axis.

color — Internal Design Entry HDL color number. The line type describes both the color and thickness of the line. When the integer is converted to a binary value, bit 0 defines the thickness (0 = *thin*), and the seven most significant bits define the color.

1 Black (default)

3 Red

5 Green

9 Blue

17 Yellow

33 Orange

67 Salmon

69 Violet

71 Brown

73 Skyblue

75 White

77 Peach

81 Pink

83 Purple

97 Aqua

99 Gray

Text

The format for text is:

```
T x y angle slant size over inv just font Nch color string
```

where

x y — Origin point for the text string.

angle — Angle of the text on the drawing:

`0, 90, 180, 270`

size — Height of the characters.

just — Justification of the added text can be:

0 Left-justified text
1 Centered text
2 Right-justified text

Nch — Number of characters and spaces in the string.

color

Internal Design Entry HDL color number. The line type describes both the color and thickness of the line. When the integer is converted to a binary value, bit 0 defines the thickness (0 = *thin*), and the seven most significant bits define the color.

1 Black (default)

3 Red

5 Green

9 Blue

17 Yellow

33 Orange

67 Salmon

69 Violet

71 Brown

73 Skyblue

75 White

77 Peach

81 Pink

83 Purple

97 Aqua

99 Gray

Note: The text definition arguments `slant`, `over`, `inv`, and `font` are not currently implemented.

Connections

The syntax for connections is:

```
C x y name dispx dispy bubbleable [default_state x2 y2 x3 y3] f size angle just
```

where

x y

Location of the connection.

name
Connection name must be enclosed in quotation marks.

dispx dispy
Location of the name.

bubbleable
Whether or not the pin can be defined as low asserted:

`0` - False

`1` - True

default_state
Whether or not the pin is low asserted:

`0` - False

`1` - True

If the default_state is `1` when a component is initially added, the pin is bubbled.

f
Whether the connection is a filled or open dot:

`0` - False

`1` - True

size
Size of the name string (default is 41).

angle
Angle of the pin name attached to the connection.

`0, 90, 180, 270`

just
Justification of the string can be:

L Left-justified text
C Centered text
R Right-justified text

Component Properties

The syntax for component properties is:

```
P name value x y angle slant size over inv just font NV VV IP color
```

where

name
Default property name must be enclosed in quotation marks.

value
Default property value must be enclosed in quotation marks.

x y
Reference point (location) of the property.

angle — Angle of the property.

`0,90,180,270`

size — Height of the characters.

just — Justification of the property can be:

0 Left-justified text
1 Centered text
2 Right-justified text

NV — Visibility of the property name:

0 Invisible
1 Visible

VV — Visibility of the property value (default is visible).

0 Invisible
1 Visible

IP — Interface property

0 Non-interface
1 Interface

color

Internal Design Entry HDL color number. The line type describes both the color and thickness of the line. When the integer is converted to a binary value, bit 0 defines the thickness (0 = *thin*), and the seven most significant bits define the color.

1 Black (default)

3 Red

5 Green

9 Blue

17 Yellow

33 Orange

67 Salmon

69 Violet

71 Brown

73 Skyblue

75 White

77 Peach

81 Pink

83 Purple

97 Aqua

99 Gray

Pin Properties

Pin properties require one line each. They are identical to component properties, except they start with an X instead of a P, and occur directly after the connection with which they are associated.

The syntax for pin properties is:

```
X name value x y angle slant size over inv just font NV VV IP color
```

where

name Default property name must be enclosed in quotation marks.

value Default property value must be enclosed in quotation marks.

x y Reference point (location) of the property.

angle Angle of the property.

`0, 90, 180, 270`

size Height of the characters.

just Justification of the property can be:

`0` - Left-justified text

`1` - Centered text

`2` - Right-justified text

NV Visibility of the property name:

`0` - Invisible

`1` - Visible

VV Visibility of the property value (default is visible).

`0` - Invisible

`1` - Visible

IP Interface property

`0` - Non-interface

`1` - Interface

color Internal Design Entry HDL color number. The line type describes both the color and thickness of the line. When the integer is converted to a binary value, bit 0 defines the thickness (0 = *thin*), and the seven most significant bits define the color.

1 Black (default)

3 Red

5 Green

9 Blue

17 Yellow

33 Orange

67 Salmon

69 Violet

71 Brown

73 Skyblue

75 White

77 Peach

81 Pink

83 Purple

97 Aqua

99 Gray

PIN_DELAY

The PIN_DELAY property values are attached to physical pins starting from 15.5.1 release. This is required only for multi-section parts as each logical pin in such parts is mapped to multiple physical pins. In the case of single-section parts or split parts, the pin number is redundant as one logical pin in these parts is mapped to a single physical pin.

The syntax for specifying PIN_DELAY for multi-section parts is:

```
PIN_DELAY ='(PinNum1:val1;PinNum2:val2 mil;PinNum4:val4)';
```

In the above syntax, `val` includes the unit also.

The following syntax is valid for specifying PIN_DELAY in the case of single-section parts and split part will also be supported for the backward compatibility:

```
    PIN_ DELAY = '(value)';
```

Bubble Groups

Bubble groups require several lines each in the symbol file. They start with a line that begins with `B` and end with a line containing only the word `END`. Each bubble group is on a line by itself in the following format:

```
[name1, name2, name3, . . .]
```

All the names are strings with quotation marks. If the bubble group is asymmetrical, the first comma is replaced by a colon.

Connectivity Design Data Files

These files, which are in ASCII format, describe all the components on a drawing, including

- Component names
- Names of signals tied to component pins (including bubble state)
- Component properties

Connectivity design data files are the files on which Design Entry HDL bases its HDL generation. Connectivity files contain complete path names to the associated library files. The system name of the symbol files is SCH with version and page number extensions (for example, <*library*>.*cell*.SCH_1. 1.csv).

Structure of Connectivity design data files

Header

Example Header in a Connectivity File

```
FILE_TYPE = CONNECTIVITY;
{CONCEPT version and date}
[expression property]
[nets]
[invokes]
END.
```

Optional sections are the

- Expression property
- Net
- Invoke

Note: The continuation character for lines in a connectivity file is a tilde (~). This character can occur anywhere in the line, even in the middle of words, but must be followed by <LF>.

Comments

Comments begin and end with braces { }. They can appear anywhere in a connectivity file except in the middle of identifiers or quoted strings and can cross lines.

Expression property on a drawing component

Example Expression Property

```
expr property ::= EXPR = expression string;
EXPR=SIZE=10;
```

NET definitions

Each time Design Entry HDL saves a connectivity file, it numbers all the nets. The NC net is always net zero. Unnamed signals are also numbered. The net numbers are not the same each time the connectivity file is written.

Example Net Definitions in a Connectivity File

```
nets ::= constant "net_name_string [property_list];
```

constant	Net number.
net_name_string	Either the signal name for the net or the unnamed signal string created by Design Entry HDL; must be enclosed in quotes.
property_list	(Optional) Property name with format: property_list ::= {identifier “string}

identifier

Property name must begin with a letter and can contain only:

- Letters
- Digits
- Underscore (_)

Each *property_list* entry must end with a newline character: FILE_TYPE and END.

“string”

The quoted string:

2UN$1$2P$A;

3ANWCLOAD37

CONNECTED_TOPAGE 4;

INVOKE commands to invoke each component in the drawing

Example Invoke Statement in a Connectivity File

```
invokes ::=
% “invoke_name_string
“version_str,xy_str,“rotation,directory_str,path_str;
[parameter_property_list];
[property_list];
{“pin_name_string [property_list] constant; }
```

invoke_name_string

Component name must be enclosed in quotes.

version_str

Symbol version number must be enclosed in quotes. This property is always output. If this property does not exist, the null string (“ “) is used.

xy_str

Coordinates of the symbol on the page. This property is always output. If this property does not exist, the null string (“ “) is used.

Vector Plot Format

This section describes the format of the plot file produced with Design Entry HDL’s *vectorize* command. This command produces an ASCII plot file that can be used to transmit drawings to other machines or that can be used to drive a pen plotter (with the aid of a format conversion program). The system name of vector plot files is vector.dat.

➤ To use the *vectorize* command, enter the command and press Return.

  vectorize creates a file named vector.dat that contains a vector plot format version of the current drawing. This file can be used to transmit files to other machines or drive a pen plotter (with the aid of a format conversion program). The vector output is a plot of the entire drawing, not just the portion showing on the screen.

There are three different types of primitives in the plot file: LINES, ARCS, and TEXT_STRINGS. The first character of the line specifies the type of the primitive. All units are nominally 0.002 inches.

Line Primitive

Lines require one line each in the file. The format is the following:

```
L x1 y1 x2 y2 [pattern] color
```

x1 y1 x2 y2 The line's endpoint coordinates; the line runs from (x1 y1) to (x2 y2). The coordinates are separated by spaces.

pattern This optional argument identifies the line style (solid, broken, and so on) as a bit pattern. For example, if pattern is -1, the line is solid and if pattern is 682, the line is dotted. See the pattern values listed with the description of wires in the ASCII file description.

```
-1

273

682

2175

3135
```

color The internal Design Entry HDL color number. The line type describes both the color and thickness of the line. When the integer is converted to a binary value, bit zero defines the thickness (0 = thin), and the seven most significant bits define the color.

1 Black (default)

3 Red

5 Green

9 Blue

17 Yellow

33 Orange

67 Salmon

69 Violet

71 Brown

73 Skyblue

75 White

77 Peach

81 Pink

83 Purple

97 Aqua

99 Gray

Arc Primitive

The format of the arc primitive is the following:

```
A x y radius start_angle stop_angle
```

x y radius The center and radius points of the arc.

start_angle/ stop_angle Floating point numbers that measure the angles, in degrees, counterclockwise from the X axis

Test String Primitive

Each text string primitive consists of the following four lines; each line is terminated by a line feed character

```
T x y

angle slant size overbar inverse_video

justification font

string
```

x y The origin point of the text string.

angle The angle of the text on the drawing. The following are allowed angles:

`0, 90, 180, 270`

justification The justification of the added text. There are three possible values:

0 The text is left justified.

1 The text is center justified.

2 The text is right justified

string The text string. No quotation marks are required.

6

The template.tsg File

You can choose *Tools – Generate View* in Design Entry HDL or run the Windows command prompt to generate symbols. You can then use the `template.tsg` file to customize these symbols, that is, set certain graphical attributes or properties and attach them to the pins or symbols.

Design Entry HDL obtains the names of pins and properties associated with symbols and pins of symbols from the source view or source file. When you run the `genviewHDL` command, it obtains information related to graphical attributes of the symbol and additional pin and symbol properties by reading a template file called `template.tsg`. Some of the attributes that you can specify using the `template.tsg` file include the following:

- Pin properties (for example, `VHDL_MODE`, `VHDL_SCALAR_TYPE`, and so on)
- Symbol properties (for example, `VHDL_GENERIC`, `LIBRARY`, and so on)
- Font size of text used in the symbol drawing
- Color of the symbol box

The default `template.tsg` file is located at `<`*`your_install_dir`*`>/share/cdssetup/concept/genview`. You can create a `template.tsg` file and place it either at your site (`$CDS_SITE/cdssetup/concept/genview)` or in your project. The `template.tsg` file is chosen according to the CSF search mechanism.

Format of template.tsg File

The sections in the `template.tsg` file and the keywords used in the sections are described below:

defcell

The `defcell` keyword must be the first keyword in the file. The remaining keywords in the file define the entire symbol. The format for `defcell` is as follows:

```
(defCell <cellname>
    [<defSymbol section>]
)
```

where `<cellname>` defines the name of the symbol being generated. For example:

```
(defcell "badder"
    ...
    ...
)
```

Note: The name of the generated symbol is obtained from the corresponding name in the source view or source file. For example, if the symbol is generated from a VHDL entity, the name of the symbol is the same as the name of the entity. The `genviewHDL` utility ignores this keyword.

defSymbol

The `defSymbol` section describes the properties and attributes of the symbol. This section is defined as follows. Click the links below for more information on the sub-sections of the `defSymbol` section.

```
(defSymbol
    [<symbolProps>]
    [<symbolParam>]
    [<symbolLabels>]
    [<pinLocSpec>]
    [<pinPosition>]
)
```

symbolProps

The `symbolProps` sub-section defines the properties along with their values that will be attached to the generated symbol. It also specifies other graphical attributes of the property name and value pair. Each symbol property can be applied to the symbol, or some or all of the pins of the symbol. This sub-section is defined as follows.

```
(symbolProps
    (defProp
        [(<property name> <property value>)]
        [(apply input|output|io|all|cellview|left|right|top|bottom)]
        [(format "off"|"value"|"name=value")]
        [(location (<expr>:<expr>))]
        [(justification left|right|center)]
        [(orientation 0|90|180|270)]
        [(fontHeight <float>)]
        [(color <colorname>)]
    )
    (defProp
        ...
        ...
    )
)
```

defProp

The `defProp` sub-section defines a property that will be attached to the generated symbol and the set of text attributes that are associated with the property. You must create a `defProp` section for each property that you want to be attached to the generated symbol. This sub-section is declared as follows:

```
    ...
    (defProp
        [(<property name> <property value>)]
        [(<text attribute 1>)]
        [(<text attribute 2>)]
        ...
    )
    (defProp
        [(<property name> <property value>)]
        [(<text attribute 1>)]
        [(<text attribute 2>)]
        ...
    )
```

Note: When working with hierarchical blocks and split symbols, ensure that the `SPLIT_BLOCK_NAME` and `SYM_NAME` properties are added to the `defProp` sub-section of the `template.tsg` file at the $CDS_SITE level.

apply

The `apply` keyword specifies the object(s) to which a property will be attached in the generated symbol. The objects can be

`cellview`	attached to the symbol
`all`	attached to all pins on the symbol
`input`	attached to input pins
`output`	attached to output pins
`io`	attached to io (inout) pins
`left`	attached to pins on the left
`right`	attached to pins on the right
`top`	attached to pins on the top
`bottom`	attached to pins on the bottom

The default value of `apply` is `cellview`.

format

The `format` keyword defines the visibility of the property—whether only the value, name and value, or neither (invisible property) will be displayed in the symbol drawing. The default is “`value`”. The possible format values are

`“off”`	Nothing is displayed.
`"value"`	Only the value is displayed.
`"name=value"`	Both the name and value are displayed.

location

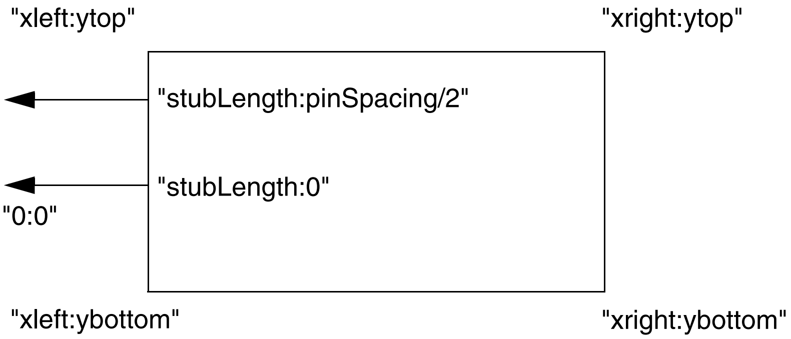
The `location` keyword defines the location where the property will be placed in the symbol. The location must be specified relative to the symbol box or symbol pins. You can define your own locations by forming simple expressions with the following constants

xleft, xright	Left and right edges of the symbol drawing
ytop, ybottom	Top and bottom limits of the symbol drawing
stubLength	Length of the pin
pinSpacing	Spacing between consecutive pins on any side

For example, you can specify the location using expressions as follows:

"xleft:ytop"	Upper left
"(xleft+xright)/2:(ytop+ybottom)/2"	Center for the symbol properties
"stubLength/2:pinSpacing/4"	Halfway down the pin stub for left pins for pin labels.

The following figure shows the various locations on the symbol, based on some defined constants.



justification

The `justification` keyword defines the justification of the properties placed at the location specified by the location keyword. The possible justification values are `left`, `center` and `right`. The default justification is left.

orientation

The `orientation` keyword specifies the orientation of the property. The possible orientation values are

`R0`	Horizontal and upright
`R90`	Vertical facing right
`R180`	Horizontal upside down
`R270`	Vertical facing left

The default orientation depends on the location of the object to which the property is attached to. If the property is attached to a pin on the top of the symbol, the default orientation is R90, whereas if the property is attached to the symbol, the default orientation is R0.

fontHeight

The `fontHeight` keyword defines a scaling factor for the height of the font used in the properties. The font value is a floating point number. The default font size used by `genviewHDL` is `0.082` inch and the scaling factor is 1.

color

The `color` keyword specifies the color that will be used for the property. The possible color values are

- red
- green
- blue
- yellow
- orange
- salmon
- violet
- brown
- skyblue

- white
- peach
- pink
- purple
- aqua
- gray

Examples of the symbolProps Section

Example 1

If you want to attach the `BLOCK=TRUE` property to every symbol you are generating so that you can use the Design Entry HDL block editor commands on the generated symbol. The following example shows the `defProp` sub-section you need to add:

```
(symbolProps
    (defProp
        (BLOCK "TRUE")
        (format "value")
        (location "(xright:ytop")
        (justification center)
        (fontHeight 0.082)
    )
)
```

This places the `BLOCK=TRUE` property at the top right side of the symbol. Only the value of the property (`TRUE`) will be visible on the symbol. Adding symbol properties using the `template.tsg` file is useful when you need to attach properties to several symbols that you will be generating using `genviewHDL`.

Example 2

```
(symbolProps
    (defProp
        (BLOCK "TRUE")
            (format  "off")
            (location "(xright+xleft)/2:ytop")
            (justification center)
            (orientation R0)
            (fontHeight 0.082)
        )
    (defProp
        (PIN_TEXT "")
            (apply all)
            (format "value")
            (fontHeight 0.066)
        )
    )
)
```

Example 3

You can use the `template.tsg` file to attach properties to a group of pins on the symbol. For example, you can attach the `VHDL_MODE=IN` property to all input pins of the symbol by adding the following `defProp` sub-section in the `template.tsg` file.

```
(symbolProps
    (defProp
        (VHDL_MODE IN)
        (location "-700:0")
        (justification left)
        (apply input)
    )
)
```

Similarly, to attach the `VHDL_SCALAR_TYPE=STD_LOGIC` property to scalar pins and `VHDL_VECTOR_TYPE=STD_LOGIC_VECTOR` to vector pins to an existing symbol, include the following section in your `template.tsg` file:

```
(symbolProps
    (defProp
        (VHDL_SCALAR_TYPE STD_LOGIC)
        (apply scalar)
    )
    (defProp
        (VHDL_VECTOR_TYPE STD_LOGIC_VECTOR)
        (apply vector)
    )
)
```

and regenerate the symbol (this can be done by editing the existing symbol and executing `genviewHDL`). `genviewHDL` attaches the `VHDL_SCALAR_TYPE=STD_LOGIC` property to scalar pins and the `VHDL_VECTOR_TYPE=STD_LOGIC_VECTOR` property to vector pins on the symbol.

symbolParam

The `symbolParam` sub-section specifies the graphical attributes of the symbol. This sub-section is defined as follows.

Click the links below for more information on the keywords used in this sub-section.

```
(symbolParam
    (origin topLeft|topRight|bottomLeft|bottomRight|center)
    (wireSpacing <float>)
    (wireLength <float>)
    (labelHeight <float>)
    (vSideLength <float>)
    (hSideLength <float>)
    (units inches|mm|cm)
    (resolution <float>)
    (<color attributes>)
)
```

origin

The `origin` keyword specifies the location where the `ORIGIN` symbol will be placed on the generated symbol. The location value can be

- topLeft
- topRight
- bottomLeft
- bottomRight
- center

wireSpacing

The `wireSpacing` keyword specifies the spacing between the pins of the symbol in user units. If this is not specified, the wire spacing is calculated from the width and height of the symbol and the number of pins. The default wire spacing is equal to the inverse of resolution.

wireLength

The `wireLength` keyword specifies the length of the pin stub and the distance from the dot to the pin name. The default wire length equals `1/`resolution.

labelHeight

The `labelHeight` keyword specifies the height of the font used for the labels or notes on the symbol. The font value is a floating point number. The default font size used by `genviewHDL` is `0.082` inch.

vSideLength

The `vSideLength` keyword specifies the length of the vertical sides of the symbol in user units. If this is not specified, the values are computed from the number of pins on the symbol and the pin spacing.

hSideLength

The `hSideLength` keyword specifies the length of the horizontal sides of the symbol in user units. If this is not specified, the values are computed from the number of pins on the symbol and the pin spacing.

units

The `units` keyword specifies the units for all the dimensions specified in the `template.tsg` file. The default value is `inches`.

resolution

The `resolution` keyword determines the round off values for the locations of pins, labels, etc. This should be normally set to the grid spacing of the symbol file. The default resolution is `0.05`.

color attributes

The following color attributes specify the color to be used while displaying the corresponding object.

Attribute	Description
`stubColor`	Color of the pin stub
`boxColor`	Color of the symbol box
`noteColor`	Color of notes on the symbol
`propColor`	Color of the properties
`pinPropColor`	Color of pin properties
`pinNoteColor`	Color of pin notes

The possible color values are

- red
- green
- blue

- yellow
- orange
- salmon
- violet
- brown
- skyblue
- white
- peach
- pink
- purple
- aqua
- gray

Example of the symbolParam Section

The following `symbolParam` sub-section:

- Places the origin of the symbol at the top left
- Generates a symbol measuring 2 x 3 (height by width in inches)
- Colors the symbol and its pins white
- Colors the notes yellow and the properties red

```
(symbolParam
    (origin topLeft)
    (vSideLength 2.0)
    (hSideLength 3.0)
    (stubcolor "white")
    (boxcolor "white")
    (pinnotecolor "yellow")
    (notecolor "yellow")
    (pinpropColor "peach")
    (propColor "skyblue")
)
```

symbolLabels

The `symbolLabels` sub-section defines the labels or notes of the symbol and the set of text attributes that are associated with the label. This sub-section is defined as follows. Click the links below for more information on the keywords used in this sub-section.

```
(symbolLabels
    (defLabel
        [(name “{<labelname>}”)]
        [(apply input|output|io|all|cellview|left|right|top|botton)]
        [(location (<expr>:<expr>))]
        [(justification left|right|center|centerRight|centerLeft)]
        [(orientation 0|90|180|270)]
        [(fontHeight <float>)]
        [(color <colorname>)]
    )
    (defLabel
        ...
        ...
    )
)
```

There are two pre-defined labels in Design Entry HDL:

- pinName
- cellName

apply

The `apply` keyword specifies the object(s) to which a label or note will be attached in the generated symbol. The objects can be

`cellview`	attached to the symbol
`all`	attached to all pins on the symbol
`input`	attached to input pins
`output`	attached to output pins
`io`	attached to io (inout) pins
`left`	attached to pins on the left

`right`	attached to pins on the right
`top`	attached to pins on the top
`bottom`	attached to pins on the bottom

location

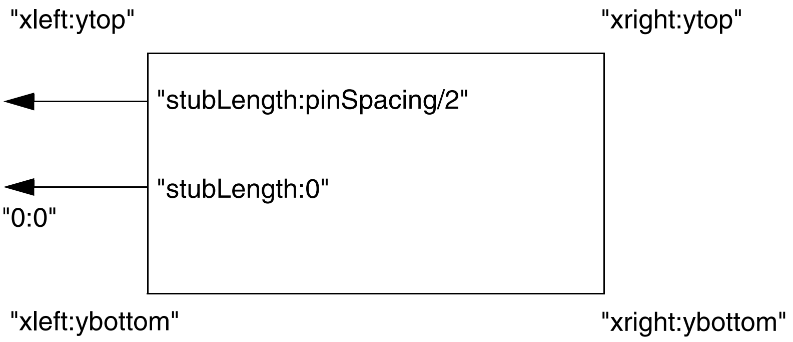
The `location` keyword defines the location where the labels or notes of the symbol will be placed in the symbol. The location must be specified relative to the symbol box or symbol pins. You can define your own locations by forming simple expressions with the following constants.

xleft, xright	Left and right edges of the symbol drawing
ytop, ybottom	Top and bottom limits of the symbol drawing
stubLength	Length of the pin
pinSpacing	Spacing between consecutive pins on any side

For example, you can specify the location using expressions as follows

"xleft:ytop"	Upper left
"(xleft+xright)/2:(ytop+ybottom)/2"	Center for the symbol properties
"stubLength/2:pinSpacing/4"	Halfway down the pin stub for left pins for pin labels.

The following figure shows the various locations on the symbol, based on some defined constants.



justification

The `justification` keyword defines the justification of the properties placed at the location specified by the location keyword. The possible justification values are `left`, `right`, `center`, `centerRight` and `centerLeft`.

orientation

The `orientation` keyword specifies the orientation of the labels or notes of the symbol. The possible orientation values are

`R0`	Horizontal and upright
`R90`	Vertical facing right
`R180`	Horizontal upside down
`R270`	Vertical facing left

The default orientation depends on the location of the object to which the label or note is attached to. If the label or note is attached to a pin on the top of the symbol, the default orientation is R90, whereas if the label or note is attached to the symbol, the default orientation is R0.

fontHeight

The `fontHeight` keyword defines the height of the font used in the labels or notes of the symbol. The font value is a floating point number. The default font size used by `genviewHDL` is `0.082` inch.

color

The `color` keyword specifies the color that will be used for the label or note. The possible color values are

- red
- green
- blue
- yellow

- orange
- salmon
- violet
- brown
- skyblue
- white
- peach
- pink
- purple
- aqua
- gray

Example of the symbolLabels Section

```
(symbolLabels
    (defLabel
        (name "Designer: Harry")
        (location "xright:ybottom")
        (orientation R0)
        (justification right)
        (fontHeight 0.082)
        (apply cellview)
    )
    (defLabel
        (name "{pinName}")
        (location "0:1.15*stubLength")
        (orientation R90)
        (justification left)
        (fontHeight 0.066)
        (apply bottom)
        (color purple)
    )
)
```

pinLocSpec

This `pinLocSpec` sub-section specifies the side of the symbol where a pin will be placed, on the basis of the name of the pin. By default, input pins are placed on the left side, output pins are place on the right side and io pins are placed at the top of the symbol. If this sub-section is specified, it will override the default pin placement. This section is defined as follows:

```
(pinLocSpec
    (rightPins <pinnames>)
    (leftPins <pinnames>)
    (topPins <pinnames>)
    (bottomPins <pinnames>)
)
```

Where `<pinnames>` specifies the names of pins whose location is being specified. The pin names are specified in the following format:

```
<pinname 1> <pinname 2>...<pinname n>
```

The `genviewHDL` utility decides the side for a pin based on its mode in the source view or source file (input, output, inout and so on) and the specification in the pinPosition sub-section. To modify the side on which a pin is placed, Cadence recommends that you modify the pinPosition sub-section.

pinPosition

The `pinPosition` sub-section specifies the side on which a pin should be placed based on the mode of its corresponding port in the source view. By default, input pins are placed on the left of the symbol, output pins are placed on the right side of the symbol, and io (inout) pins are placed on the top of the symbol. This sub-section is defined as follows:

```
(pinPosition
    (input <side>)
    (output <side>)
    (io <side>)
)
```

where `<side>` can be `left`, `right`, `top` or `bottom`.

Example of the pinPosition Section

To place input pins at the left and output and io pins to the right, add the following pinPosition sub-section in the `template.tsg` file:

```
(pinPosition
    (input left)
    (output right)
    (io right)
)
```

Pin Name and Cell Name Replacement

Because the `template.tsg` file can be used to attach properties to a group of pins, you might want to use the name of the pin in these properties. For example, you might want to attach the property `<p inname>_DELAY` to all the io pins on the symbol. The `template.tsg` file allows you to specify the name of the property as `{pinName}_DELAY` (applied to io pins). When the property is generated, `genviewHDL` replaces the string `{pinName}` with the actual pin name. Consequently, if there are two io pins, `io1` and `io2`,the `io1_DELAY` property will be attached to the `io1` pin, and the `io2_DELAY` property will be attached to the `io2` pin.

Similarly, if you want to attach a property `<design name>_TYPE` to all your symbols when they are generated, insert a `defProp` sub-section with the property name as `{cellName}_TYPE`. When `genviewHDL` generates a symbol named `badder`, it will attach the property `badder_TYPE` to it.

Sample template.tsg File

The following `template.tsg` file illustrates the use of some of the sections described earlier.

```
(defCell ""
    (defSymbol symbol
    (symbolLabels
        (defLabel
            (name "{cellName}")
            (location "(xright+xleft)/2:(ytop+ybottom)/2")
            (orientation R0)
            (justification center)
            (fontHeight 0.082)
            (apply cellview)
    )
        (defLabel
            (name "{pinName}")
            (location "0:1.15*stubLength")
            (orientation R90)
            (justification left)
            (fontHeight 0.082)
            (apply bottom)
        )
        (defLabel
            (name "{pinName}")
            (location "0:-1.15*stubLength")
            (orientation R90)
            (justification right)
            (fontHeight 0.082)
            (apply top)
        )
        (defLabel
            (name "{pinName}")
            (location "0:-1.15*stubLength")
            (orientation R00)
            (justification centerRight)
            (fontHeight 0.082)
            (apply right)
        )
        (defLabel
            (name "{pinName}")
            (location "0:1.15*stubLength")
            (orientation R0)
            (justification centerLeft)
            (fontHeight 0.082)
            (apply left)
        )
    )
    (symbolParam
        (origin center)
            (wireSpacing 0.2)
            (wireLength 0.1)
            (labelHeight 0.082)
            (stubColor "green")
            (boxColor "green")
            (pinnotecolor "orange")
            (notecolor "yellow")
            (pinpropColor "peach")
            (propColor "skyblue")
        )
        (pinPosition
            (input left)
            (output right)
            (io top)
        )
    )
)
```

The following notes briefly describe the sample `template.tsg` file:

The name of the symbol:

- Appears at the center of the symbol
- Is justified center
- Is horizontal
- Has a font height of `0.082` inches (the default unit used is inches)

The pins on the bottom of the symbol are labeled. The label for each pin:

- Appears at (`0, 1.15*stubLength`), and the location of the connection is (`0,0`). The label appears slightly above the bottom of the symbol border.
- Is vertically oriented
- Is justified left
- Has a font height of `0.082` inches

The labels for pins on the top, left, and right are defined similarly.

The origin of the symbol will be at the center. The spacing between pins will be `0.2`, and the length of each pin will be `0.1` inches. The font height of the labels and properties associated with the symbol will be `0.082` inches. The pin and the symbol will be green. The pin notes will be orange. The symbol notes will be yellow. The color of the pin properties will be peach, and the symbol properties will be sky blue in color.

The input pins (the mode is obtained from the source view, and an input pin corresponds to an input port in the source view) will be placed on the left of the symbol, the output pins will be placed to the right, and the io pins will be placed at the top of the symbol.

7

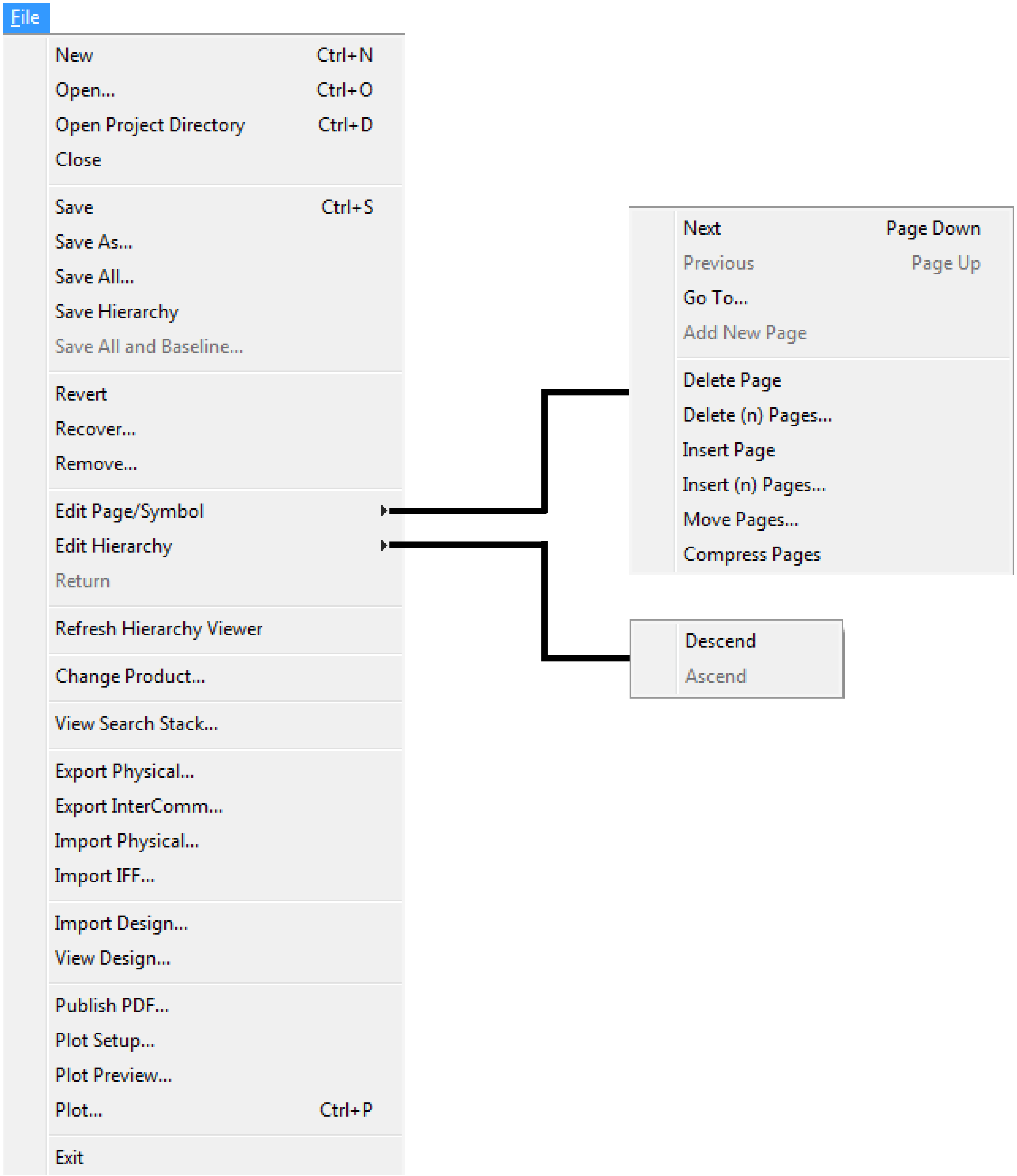
Menu Help

Overview of Design Entry HDL Menus

This section describes the various menus and submenus in the Allegro Design Entry HDL schematic editor.

- ❑ File Menu
- ❑ Edit Menu
- ❑ View Menu
- ❑ Component Menu
- ❑ Wire Menu
- ❑ Text Menu
- ❑ Block Menu
- ❑ Group Menu
- ❑ Variants Menu
- ❑ Display Menu
- ❑ PSpice Simulator Menu
- ❑ RF-PCB Menu
- ❑ Design Management Menu
- ❑ Tools Menu
- ❑ Window Menu
- ❑ Help Menu

File Menu


File
New Ctrl+N
Open... Ctrl+O
Open Project Directory Ctrl+D
Close
Save Ctrl+S
Save As...
Save All...
Save Hierarchy
Save All and Baseline...
Revert
Recover...
Remove...
Edit Page/Symbol
Edit Hierarchy
Return
Refresh Hierarchy Viewer
Change Product...
View Search Stack...
Export Physical...
Export InterComm...
Import Physical...
Import IFF...
Import Design...
View Design...
Publish PDF...
Plot Setup...
Plot Preview...
Plot... Ctrl+P
Exit
Next Page Down
Previous Page Up
Go To...
Add New Page
Delete Page
Delete (n) Pages...
Insert Page
Insert (n) Pages...
Move Pages...
Compress Pages
Descend
Ascend

File – New

Procedure Command

Creates a new drawing. Drawings are named UNNAMED.SCH.1.1 until you save the drawing under another name. The 1.1 designators are for version and page.

File – Open

Procedure Command

Use this menu option to open an existing drawing. Opens the *View Open* dialog box to specify a library, cell, and view (drawing) to edit.

Note: You can open the same drawing in more than one window. Editing the same design this way lets you look at different views of the same drawing. You can:

- ❑ Zoom the windows independently to focus on different sections of the design.
- ❑ Use *Wire – Draw* and *Wire – Route* to connect points between windows.
- ❑ Use the new window as a global view of the original drawing.

File – Open Project Directory

Use this menu option to open the folder where the schematic, which is currently open in the schematic canvas, is stored.

File – Close

Procedure Command

Closes the current drawing. If you want to save changes to the drawing, choose File – Save before *File – Close*.

Note: You cannot close the Design Entry HDL main window using *File – Close*.

Choose File – Exit to exit Design Entry HDL. A prompt appears asking you if you want to save or discard changes in the drawing, even if they are not visible in a window.

File – Save

Procedure Command

Saves the drawing. If you save the drawing with the *Check On Write* (under *Tools – Options—Check*) or *Confirm Write* (under *Tools – Options—Output*) setup options turned on, errors in the design are recorded in the `cp.mkr` and `netlister.mkr` marker files in your temporary directory.

You can also do the following:

- Save the drawing with another name using File – Save As
- Revert to the last saved version using File – Revert
- Choose Display – Modified to list the files that were modified but not saved during the current session.
- Change default settings for save confirmation using Tools – Options

File – Save As

Procedure Command

Allows you to save the current drawing with a new name in any library (directory)

File – Save All

Saves all the currently open drawings that have been modified.

File – Save Hierarchy

Reads all the pages in a hierarchical design and saves them.

File – Save All and Baseline

Displays the Baseline dialog box. Use this dialog box to save and baseline a schematic. This menu option is enabled only if the Generate Schematic Metadata check box is selected in the Metadata Options page of the Design Entry HDL Options dialog box.

For more information on baselining, see the *How Baselining Works* topic in *Allegro Design Entry HDL User Guide*.

File – Revert

Procedure Command

Replaces the current drawing with the last-saved version. Choose *File – Save As* to save the drawing with a new name.

File – Recover

Procedure Command

Recovers drawings that were being edited if Design Entry HDL or your system crashes.

Whenever you start Design Entry HDL, a temporary directory is created in the <*project_directory*>/temp directory. By default, `xxnedtmp` is the name of the temporary directory. If the `xxnedtmp` directory already exists, a `xxnedtmp1` directory is created. If these two directories already exist, `xxnedtmp2` is created, and so on. An undo log file for each drawing is stored in this directory. The name of the undo log file for the first drawing edited is undo1.log. The second drawing's undo log file is `undo2.log`, and so on.

Select the undo log file for the drawing you want to recover and click *Open*.

Design Entry HDL gives the recovered drawing a unique name (for example, `RECOVER1.SCH.1.1`). Recovered drawings are only saved in memory, not on disk. Choose File – Save As to save the drawing with a new name.

File – Remove

Command

Deletes a specified cell, view, or file from disk.

File – Edit Page/Symbol – Next

Procedure Command

Displays the next page of a multipage drawing.

If you are viewing the symbol for a component, Design Entry HDL displays the next symbol.

File – Edit Page/Symbol – Previous

Procedure Command

Displays the previous page of a multipage drawing.

If you are viewing the symbol for a component, Design Entry HDL displays the previous symbol.

File – Edit Page/Symbol – Go To

Procedure Commands

Displays the *Go To Page/Symbol* dialog box, which lets you specify the page of a multiple page drawing to be viewed.

To go to a specific page in a hierarchical design, select the *Calculate page number in hierarchy* check box, enter the page number and click *OK*. If you do not select the *Calculate* *page number in hierarchy* check box, you can only go to a page within the cell in which the currently open schematic page exists.

Commands

- Edit
- Gotosheet

File – Edit Page/Symbol – Add New Page

Inserts a new page in Design Entry HDL if no page exists after the current page.

File – Edit Page/Symbol – Delete Page

Deletes the current page in Design Entry HDL.

File – Edit Page/Symbol – Delete(n) Pages

Deletes a set of page in Design Entry HDL.

File – Edit Page/Symbol – Insert Page

Inserts a new page in Design Entry HDL at the current location.

File – Edit Page/Symbol – Insert(n) Pages

Inserts a set of pages at a specified location in Design Entry HDL.

File – Edit Page/Symbol – Move Pages

Moves a set of pages at a specified location in Design Entry HDL.

File – Edit Page/Symbol – Compress Pages

Removes a set of blank pages in Design Entry HDL.

File – Edit Hierarchy – Descend

Procedure Command

Descends the drawing hierarchy. File – Edit Hierarchy – Ascend returns you to the original view. Choose File – Return to return to the previous drawing in the hierarchy.

To descend into drawings while in In Hierarchy mode, you can set the environment variable CONCEPT_DESCEND_EDIT_LIST.

Example:

IF you have `vlog_rtl`, `sch_1`, and `sym_1` views of the drawing and you wish to descend into them when you double-click on the top-level drawing, set the following environment variable:

```
Setenv CONCEPT_DESCEND_EDIT_LIST vlog_rtl, sch_fs1, sym_1
```

After setting this environment variable, when you double-click on the drawing, Design Entry HDL searches for the `vlog_rtl` view and displays it. If this view is not present, Design Entry HDL displays the `sch_1` view.

File – Edit Hierarchy – Ascend

Procedure

Ascends the drawing hierarchy. Choose File – Edit Hierarchy – Descend to traverse the drawing hierarchy one level lower. Choose File – Return to return to the previous drawing in the hierarchy.

File – Return

Procedure Command

Returns you to the drawing that was previously edited in the same window.

Note: For each window, Design Entry keeps a list of drawings in the *Window* menu that were edited in that window.

File – Refresh Hierarchy Viewer

Updates the tree structure in the Hierarchy Viewer window with any changes made to the design, such as deleting or adding a new module to the design.

File – Change Product

Displays the *Product Choices* dialog box, which allows you to select the product suite whose license you want to use.

File – View Search Stack

Procedures Command

Displays the *Search Stack* dialog box, which you use to specify the library search order. Design Entry HDL searches libraries in the order that they appear in the Search Stack.

The library listed on top is checked first. If a component is not found in that library, the next library is searched and so on through the search stack. There is no limit to the number of libraries that can be used at one time.

Note: The Search Stack is only used if you are in command mode and enter the `add` component command in the console window. Components are added according to the library search order as specified in the Search Stack. If you choose Component – Add and use Part Information Manager to add components, you always fully specify the location of a component.

File – Export – Export Physical

Displays the *Export Physical* dialog box for specifying the board layout to be updated with schematic data. For further information on exporting schematic data to your board layout, see *Design Synchronization help*.

File – Export – Export InterComm

Creates and saves an InterComm-specific output file that can be loaded into the InterComm tool. The InterComm-specific IFF file is generated by a script associated to the Export InterComm option.

File – Import – Import Physical

Displays the *Import Physical* dialog box where you specify the board file from which the feedback files will be generated to update the schematic. For further information on importing physical design data to your schematic design, see *Design Synchronization help*.

File – Import – Import IFF

Displays the *IFF Import* dialog box. Use this dialog box to import designs in the Intermediate File Format (IFF) into Design Entry HDL.

IFF (Intermediate File Format) is used to transfer a design in machine and application independent format between Electrical Engineering design and Printed Circuit Board (PCB) design environments.

Design Entry HDL supports the import of this file in two ways:

- Importing IFF as a new project
- Importing IFF into an existing project

File – Import – Import Design

Displays the *Import Design* dialog box. Use this dialog box to specify the `project.cpm` file of the project or the `cds.lib` file associated with the project from which you want to import a sheet. You can import one or more schematic sheets to the currently open project from another project.

File – View Design

Spawns a read-only instance of Design Entry HDL Viewer to let you browse a design even when you are working on another design. You can perform the following activities in the read-only viewer:

- View the hierarchy of the project.
- Descend to a particular block/sheet.

File – Publish PDF

Displays the *Publish PDF* dialog box to publish a schematic design as a PDF document.

For more information on the Publish PDF utility, see the *Allegro Design Publisher User Guide*.

File – Plot Setup

Procedure Command

Displays plot setup options in the *Design Entry Options* dialog box.

File – Plot Preview

Procedure

Plots the drawing to a window on the screen so you can check scaling and other options before plotting to your printer.

File – Plot

Procedure Commands

Displays the *Print* dialog box in which you specify the printer, all or portions of the current drawing to plot, the number of copies, and other print options.

To specify default plot settings, choose Tools – Options.

Note: Regardless of default plot settings, you can print screen contents only by specifying Selection in the *Print* dialog box.

Commands

Console Commands on WIndows

Command	Capability
plot	Plots currently opened drawing
plot cache	Plots all pages of cache.
plot cache.sym.1.1	Plots symbol view of cache
plot cache.sym.1.2	Plots page 2 schematic of cache
plot cache.sch.1.*	Plots all pages of version 1

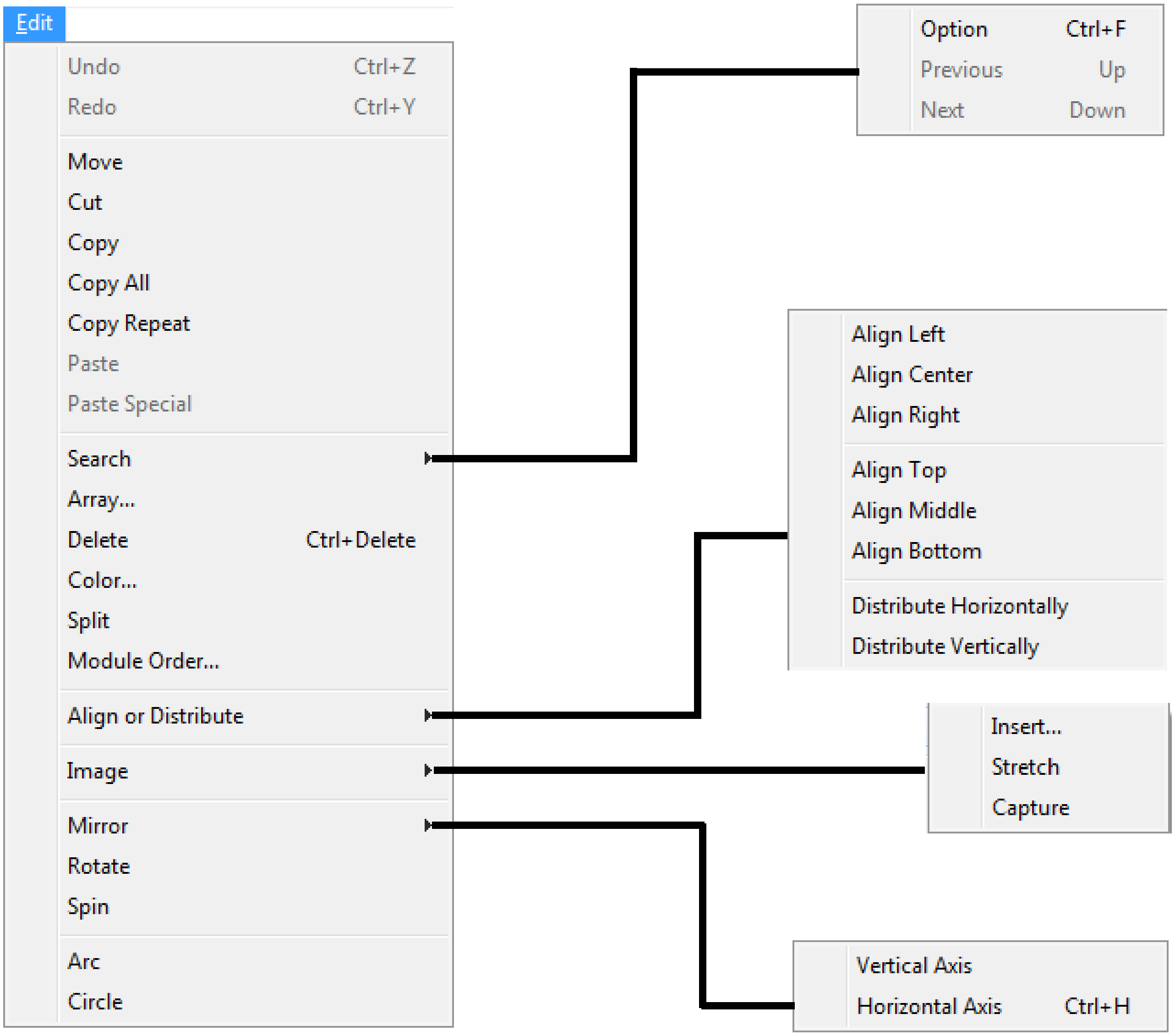
Where cache is the name of the drawing.

File – Exit

Procedure Command

Exits Design Entry HDL. Design Entry HDL prompts you to save or discard changes in opened drawings that have not been saved. Choose File – Close if you want to close a window but not exit Design Entry HDL.

Edit Menu



Edit – Undo

Procedure Command

Undoes the last action and affects only the current drawing. Repeatedly choosing *Edit – Undo* reverses previous operations performed during the editing session. Choose Edit – Redo to reverse an undo operation.

Edit – Redo

Procedure Command

Reverses the previous Edit – Undo. Repeatedly choosing *Edit – Redo* reverses previous undoes performed by *Edit –Undo*.

Edit – Move

Procedures Command

Moves objects in the current drawing or between drawings in different windows. Choose Group – Move [A] to move groups of objects.

Note: If you want to move objects between drawings and then want to undo the operation, you must choose Edit – Undo once in each of the drawings.

Use the pop-up menu to automatically change attachments for a component you are moving to another location in the design.

Procedures

- Moving text, wires, or an unwired component
- Moving a wired component
- Moving multiple objects

Edit - Cut

Procedure Command

Cuts an object (without its properties) in the current drawing or between drawings in different windows. Use the pop-up menu to retain the selected object and cut it in several unrelated places on the drawing without having to re-select the object (*Retain*/*Terminate Selection*).

You cannot cut

- ❑ default symbol properties without the associated component
- ❑ PATH properties
- ❑ soft properties
- ❑ PN properties
- ❑ properties generated by

  Tools – Back Annotate

  Component – Section – Single Section

  Component – Swap Pins
- ❑ signal names
- ❑ wire properties

Note: If you want to cut objects between drawings and then want to undo the operation, you must choose Edit – Undo once in each of the drawings.

Edit – Copy

Procedure Command

Copies an object (without its properties) in the current drawing or between drawings in different windows. You can also copy groups of objects, excluding properties, by choosing Group – Copy [A]. Use the pop-up menu to:

- ❑ *Copy All* properties (section properties, soft properties, pin properties, wire properties, and properties attached to each other) with the object. This is useful if you are copying a section of logic from one drawing to another. You can also choose Edit – Copy All to copy an object with its properties.
- ❑ Retain the selected object and copy it in several unrelated places on the drawing without having to re-select the object (*Retain*/*Terminate Selection*).

You cannot copy

- ❑ default symbol properties without the associated component
- ❑ PATH properties
- ❑ soft properties
- ❑ PN properties
- ❑ properties generated by

  Tools – Back Annotate

  Component – Section – Single Section

  Component – Swap Pins
- ❑ signal names
- ❑ wire properties

Choose Edit – Copy All to copy an object and its properties, or choose Edit – Array to create multiple copies of an object.

Edit – Copy All

Procedure Command

Copies an object and its properties in the current drawing or between drawings in different windows. This is especially useful if you are copying a section of logic from one drawing to another. You can also copy groups of objects with their properties by choosing Group – Copy All [A].

Note: Copy All copies all properties except the PATH property.

When copying a property, you indicate whether you want to attach the property to a component, pin, or wire.

Use the pop-up menu to retain the selected object and place several copies of it without having to re-select the object (*Retain*/*Terminate Selection*).

Choose Edit – Copy to copy an object without properties, or choose Edit – Array to create multiple copies of an object.

Edit – Copy Repeat

Command

Copies an object when you click on it once and pastes the object every time you click again.

Edit – Paste

Pastes an object (without its properties) in the current drawing or between drawings in different windows.

Edit – Paste Special

Procedure

Displays the Paste Special dialog box. You use this dialog box to specify whether you want to paste copied schematic parts on to the target schematic directly or change the signal names of the schematic before pasting them on the target location.

Edit – Search – Option

Displays the Find dialog box, which you can use to limit the scope of search. On the schematic page, searches are performed based on your selected objects, or, based on selected filter options such as components, nets, properties, notes, images, pins, and plumbing bodies.

You can customize searches on the current page, design or a schematic selection. You can search by using wildcards, property names, and values. You can use the Search toolbar to find objects and open the Find dialog box:

See the *Finding Design Objects* section in Allegro Design Entry HDL User Guide for details.

Edit – Search – Previous

Seaches for the previous instance of your search item.

Edit – Search – Next

Seaches for the next instance of your search item.

Edit – Array

Procedure Command

Makes multiple copies of an object in the current drawing or between drawings in different windows. You can also make multiple copies of a group by choosing Group – Array [A].

Use the pop-up menu to:

- ❑ *Copy All* properties (section properties, soft properties, pin properties, wire properties, and properties attached to each other) with the object. This is useful if you are copying a section of logic from one drawing to another. You can also choose Edit – Copy All to copy an object with its properties.
- ❑ Retain the selected object and copy it in several unrelated places on the drawing without having to re-select the object (*Retain*/*Terminate Selection*).

You cannot copy

- ❑ Signal names
- ❑ Wire properties
- ❑ Groups of properties (properties attached to objects are copied with the group)

Choose Edit – Copy to copy an object without properties, or choose Edit – Copy All to copy an object with its properties.

Edit – Delete

Procedure Command

Deletes an object from the drawing. To delete a group of objects, choose Group – Delete [A].

You cannot delete default properties.

Edit – Undo lets you retrieve deleted objects.

Edit – Color

Procedure Command

Displays the color toolbar. When the toolbar is visible, you can select a color on it, and then select objects to be changed to that color.

You can also select colors for groups of objects (Group – Color [A]). Use Tools – Options to establish default colors for objects and window background.

On a monochrome display, use *Display – Color* to view the current color of objects.

Edit – Split

Procedure Command

Splits a wire or separates two or more overlaid objects. The objects that you select blink momentarily.

Edit – Module Order

Prompts you to use the Hierarchy Viewer window for performing module ordering functions, if the Hierarchy Viewer window is not already open, it is opened.

Edit – Align or Distribute

Align or distributes objects, either vertically or horizontally. The Align function aligns a selected set of objects with respect to a common axis. The Distribute function equally spaces a group of objects according to the type of distribution—horizontal or vertical.

- Align Left
- Align Center
- Align Right
- Align Top
- Align Middle
- Align Bottom
- Distribute Horizontally
- Distribute Vertically

See Alignment and Distribution in *Allegro Design Entry HDL User Guide* for more information.

Edit – Image – Insert

Inserts an image (.bmp or .jpeg format) in the schematic canvas from an external location.

Edit – Image – Stretch

Stretches a selected image horizontally or vertically on the schematic. You can also specify the starting and ending coordinates for stretching the image.

Edit – Image – Capture

Captures screen shots of a selected part on a schematic. When you capture an image, it is copied to the clipboard from where it can be pasted into any graphics editor or a graphics-aware text editor such as Microsoft Word.

Edit – Mirror

Command

Creates a mirrored version of a component or block. If editing a symbol drawing, not all lines and arcs in the drawing are mirrored. When mirrored, justified text is shifted from left-to-right or right-to-left. You can also rotate (Edit – Rotate), spin (Edit – Spin), or replace a component with its next version (Component – Version).

Caution

Mirroring components that have unmarked pins (for example, pass-through pins and some merge symbols) reverses the bits and can result in subtle design errors.

Edit – Mirror – Vertical Axis

Command

Creates a mirrored version of a selected symbol along the vertical axis (y-axis). To create a mirrored version of a symbol, choose this menu option and select the symbol with the left mouse button.

Edit – Mirror – Horizontal Axis

Command

Creates a mirrored version of a selected symbol about horizontal axis (x-axis). To create a mirrored version of a symbol, choose this menu option and select the symbol with the left mouse button.

Edit – Rotate

Command

Rotates components and text strings 90 degrees with mirrors at 180 and 270 degrees. When a symbol is rotated, all notes and properties are also rotated and translated. You can then act on the properties independently.

When you select a component to rotate, you can repeatedly click to rotate the component another 90 degrees. You can also spin (Edit – Spin) or mirror (Edit – Mirror) a component.

Caution

Rotating some parts 180 degrees reverses the order of the pins. This can cause subtle errors in your designs if pins become incorrectly wired. To avoid this, a 180 degree rotation of a part becomes a mirror of a 0 degree rotation (about the X axis). A 270 degree rotation of a part is a mirror of a 90 degree rotation (about the x axis). To get the other two rotations and the other two mirrors, use the MIRROR command to create.

Edit – Spin

Command

Changes the orientation of components and text strings in 90-degree increments (0, 90, 180, 270). After you spin a component, you can work with its properties individually. *Edit – Spin* performs a true rotation of a component, as opposed to *Edit – Rotate*, which combines a mirror and rotation.

You can also rotate (Edit – Rotate) or mirror (Edit – Mirror) a component.

Caution

Spinning components that have unmarked pins (for example, pass-through pins and some merge symbols) reverses the bits and can result in subtle design errors.

Edit – Arc

Procedure Command

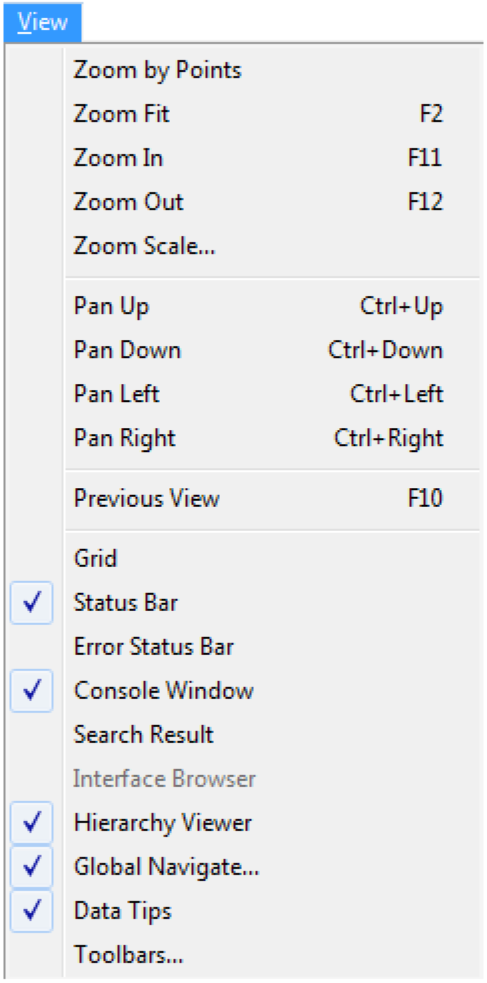
Creates an arc, typically for use in symbol drawings. You can also create a circle from an arc, or you can choose Edit – Circle.

Edit – Circle

Procedure Command

Creates circles, typically for use in symbol drawings. See also Edit – Arc. Use a circle instead of a wire to represent a low-asserted (bubbled) pin. The signal name should also be low-asserted.

View Menu



View – Zoom by Points

Procedure Command

Zooms in on an area that you define by specifying two diagonal points (opposite corners of a rectangle).

View – Zoom Fit

Procedure Command

Fits the entire drawing within the Design Entry HDL window.

View – Zoom In

Procedure Command

Enlarges the size of the drawing incrementally for up-close viewing.

View – Zoom Out

Procedure Command

Reduces the size of the drawing incrementally to let you see more of it.

View – Zoom Scale

Procedure Command

Enlarges or reduces the size of the drawing by a scale factor. The center of the window remains constant.

Use whole numbers to enlarge the drawing; use a decimal to reduce the drawing. For example, 2 zooms in by a factor of 2, and .5 zooms out by a factor of 2.

View – Pan Up

Procedure Command

Lets you view the top portion of the drawing.

View – Pan Down

Procedure Command

Lets you view the lower portion of the drawing.

View – Pan Left

Procedure Command

Lets you view the left side of the drawing.

View – Pan Right

Procedure Command

Lets you view the right side of the drawing.

View – Previous View

Command

Returns the screen display to the previous window scale and position.

View – Grid

Procedure Command

Turns the grid on or off in the currently displayed drawing.

View – Status Bar

Procedure

Shows or hides the status bar. The status bar provides feedback information from Design Entry HDL.

View – Error Status Bar

Procedure

Shows or hides the Markers status bar. The Markers status bar tells you how many errors were found when you run a check with Tools – Check.

View – Console Window

Procedure

Shows or hides the console window, where you can enter commands and view Design Entry HDL messages.

View – Search Result

Shows/hides the Search Result pane in the Component window.

View – Interface Browser

Opens the Interface Browser. While capturing a logical design in DE-HDL, net groups are created using the Interface Browser user interface. For details about using the Interface Browser, see the *Working With Net Groups and Port Groups* guide.

View – Hierarchy Viewer

Shows/hides the Hierarchy Viewer window.

View – Global Navigate

Procedure

Shows/Hides the *Global Navigation* window, which lets you select a net and view all synonyms of that net across a multipage schematic or hierarchical design.

You can also use this window to cross-probe other tools, such as Allegro or SPECCTRAQuest. If the *Global Navigation* window is visible when you have another tool open, and you select a net or cell in another tool, all instances of the selected object are listed in the status area box of the *Global Navigation* window. You can then select any item in the list to view it in your schematic design.

View – Data Tips

Enables or disables small tips on objects of a schematic, which are displayed when you place the cursor on the object.

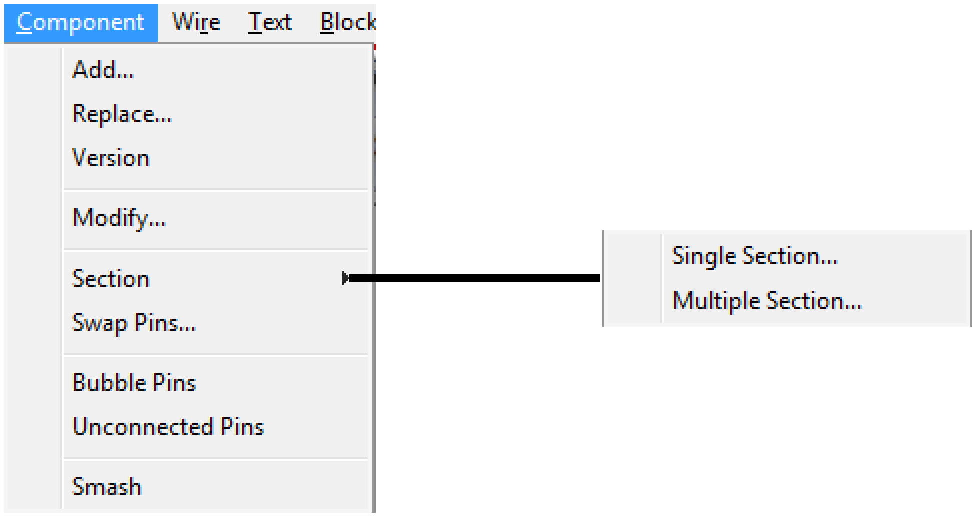
View – Toolbars

Procedure

Shows or hides the specified toolbars:

Standard	Displays toolbar shortcuts for standard file and drawing functions.
Add	Displays toolbar shortcuts for adding various objects to a drawing.
Block	Displays toolbar shortcuts for block operations.
Markers	Displays toolbar shortcuts for checking errors in a schematic.
Edit	Displays toolbar shortcuts for commands used in schematic editing.
Group	Displays toolbar shortcuts for object grouping functions.
Color	Displays the Design Entry HDL color palette.

Component Menu



Component – Add

Dialog Box   Command

Adds a component to a drawing. You choose from a library view or a category view and specify whether you want to add the component in logical or physical mode. In logical mode, only the symbol is added to the drawing. In physical mode, you can further define the physical attributes of the component. By default, Design Entry HDL adds version 1 of components.

When you make your selection, a copy of the component is attached to the cursor for you to place on the drawing. With the component attached, you can use the pop-up menu to do the following:

- ❑ Name the component
- ❑ Rotate the component
- ❑ Place a version other than version 1 of the component

You can also display Part Information Manager from the pop-up menu.

Component – Replace

Procedure Command

Replaces one component with another. By default, Design Entry HDL replaces components with version 1 of the new component.

If you are in the pre-select mode in Design Entry HDL, you can replace multiple components by doing the following:

1. Use Ctrl+click or SHIFT+click to select components one after another.
2. Choose *Component – Replace* to display Part Information Manager.
3. Select the component that should replace all the components.

When you replace a component:

- ❑ Property names and default values are retained, except those generated from

  Tools – Back Annotate

  Component – Section – Single Section

  Component – Swap Pins
- ❑ Default properties with a placeholder value (? value) receive the value of the property with the same name on the replaced component if one exists.
- ❑ Wire connections are retained if the pins are in the same location as on the original components.
- ❑ Unnamed wires are deleted.
- ❑ Rotation is preserved.
- ❑ Pin properties are reattached if pin names are the same as on the original components. Unmatched pin names cause pin properties to become symbol properties.

Component – Version

Procedure Command

Replaces a component with its next version. Design Entry HDL determines the current version of a component and displays the next version in sequence. The separate versions of a component must all reference the same schematic. Using a different version of a component has no influence on the logic drawing that defines it.

Component – Modify

Procedure Command

Lets you modify the physical properties of a component. When you select a part, a pop-up list of package types appears. You select a package type to display the Physical Parts Filter dialog box, which you use to include or change physical information from a Physical Part Table (PPT).

Component – Section – Single Section

Procedure Command

Assigns a physical section to a logical component and displays different pin numbers for the different sections. You can section a component either before or after you package the design. When you change section assignments after importing physical data, you assign just the sections you want to force and leave the others. The schematic may then have some duplicate section numbers. When you repackage the drawing, duplicate sections are reassigned.

Sectioning a part automatically assigns path properties to the drawing. You can only section components with SIZE = 1 or HAS_FIXED_SIZE characteristics.

Component – Section – Multiple Sections

Assigns physical sections to multiple logical part instances and displays different pin numbers for different sections depending on the section increment you specify.

Component – Swap Pins

Procedure Command

Swaps two pins that are part of the same pin group and that are defined in the library as swappable. Also swaps sections within components defined as HAS_FIXED_SIZE. You can only swap pins after initially assigning pin numbers using *Component – Section*.

Component – Swap Pins attaches can be deleted and moved but not changed. Once pins on a part have been swapped, the part cannot be resectioned.

Component – Bubble Pins

Procedure Command

Switches the state of a pin between bubbled (active high) and un-bubbled (active low). If the pins are part of a bubble group, you can use *Component – Bubble Pins* to convert the component from one form to another.

Component – Unconnected Pins

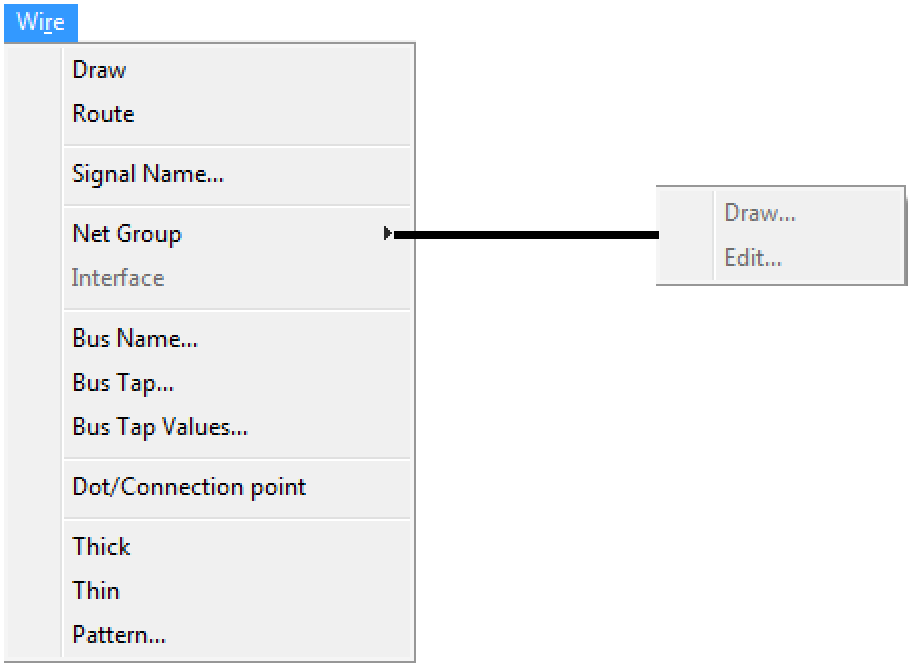
Acts as a toggle for showing or hiding unconnected pins on components. All the unconnected pins on the components of a drawing are marked with pink dots.

Component – Smash

Procedure Command

Breaks a component into its individual elements. Any properties attached to the component are deleted. This is useful for creating library component drawings. *Text – File* performs a similar operation with text.

Wire Menu



Wire – Draw

Procedure Command

Draws a wire that you manually route around objects. Wires can be orthogonal or direct depending on the setup option you choose. Wires are automatically terminated at pins and wire junctions. Double-click to end a wire that is not attached to a pin or to another wire.

To automatically route a wire around objects, choose Wire – Route.

To wire blocks, choose Block – Draw Wire or Block – Route Wire.

Use the pop-up menu to do the following:

- ❑ Change the wire orientation (*Orientation*).
- ❑ Name the wire (*Signal Name*).

Choose Tools – Options to change the default settings for wires.

Wire – Route

Procedure Command

Draws a wire, automatically routing it around objects in its path. If Design Entry HDL cannot determine a route, a diagonal line is drawn directly between the blocks.

To manually route a wire around objects, choose Wire – Draw. To wire blocks, choose Block – Draw Wire or Block – Route Wire.

Wire – Signal Name

Procedure Command

Displays the *Signal Name* dialog box where you enter one or more signal names and then select the wires you are naming.

Design Entry HDL handles signal names as properties. When editing a symbol drawing, signal names are called PN properties and can only be attached to pin connections.

Wire – Net Group – Draw

Draws a net group object. To instantiate a net group, you need to draw it on the schematic.

Note: In a design, there can only be one instance of a net group. This implies that a net group with the same name drawn on multiple schematic pages will be treated as a single net group.

For details, see the Instantiating a Net Group section in the Net Groups chapter of the *Working With NetGroups and PortGroups* guide.

Wire – Net Group – Edit

Allows you to edit net groups on the schematic canvas. For details, see the Removing Net Group Members section in the Net Groups chapter of the *Working With NetGroups and PortGroups* guide.

Wire – Bus Name

Procedure Command

Names signals on a bus that has been broken out into individual bits. This is especially useful when working with large buses whose names differ only in bit subscripts. (Alternatively, you would choose Wire – Signal Name to name each wire separately.) You specify a bus name, the most significant bit, the least significant bit, and the increment.

Wire – Bus Tap

Command

Adds a tap from a bus to a pin.

When you add bus taps, the order in which you choose the bus tap and the pin matters. To add a bus tap, you need to tap the bits from a bus wire to the dot connection point of the pin. To add a bus tap, follow these steps:

1. Choose *Wire – Draw* to draw a wire.
2. Choose *Wire – Signal Name* to add the signal name, such as `A<12..0>`.
3. Choose *Add – Bus Tap*
4. Click the bus wire to create the bus tap.
5. Extend the wire from the added tap to the dot connection point of the pin.

Important

> Do not connect the wire coming out of the bus tap to an already connected wire to the pin.

Note: If you do not enter a bus tap value, a `?` is used for the tap bits.

Note: In the Graphics page of the Design Entry HDL Options dialog box (*Tools – Options),* select the *Auto Name on Tap* check box to create taps with automatically numbered bits and named wires.

Wire – Bus Tap Values

Procedure   Command   Example

Attaches values to bus taps, numbering a range of bus taps automatically. This is especially useful when working with large buses. Alternatively, you would choose Text – Change to specify the values of a bus tap.

Wires – Bus Tap Values adds the BN property to the bus tap.

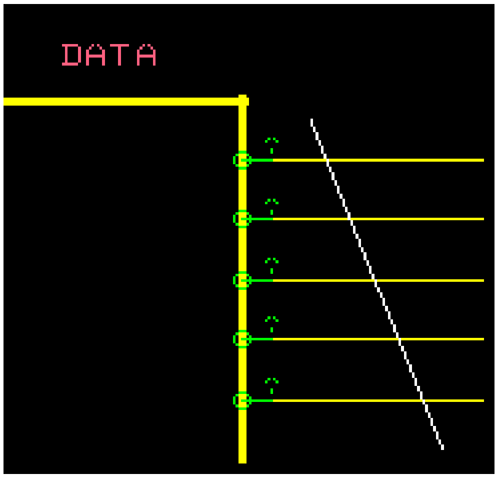
Example

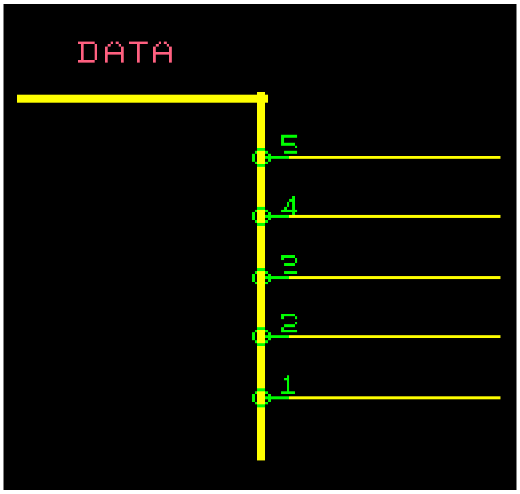
Say you want to give values for 5 taps on a wire. You might specify:

- ❑ 5 as the most significant bit
- ❑ 0 as the least significant bit
- ❑ 1 as the increment

This input results in bit number values of **<5>** through **<1>**, with **<5>** placed on the tap closest to the first location you click and <1> on the tap closest to the second location you click.

In this example, the first click is above the top bit.





The second click is below the bottom bit. Design Entry HDL draws a line between the two points.	Then the bus names and values appear.

Wire – Dot/Connection Point

Procedure Command

Adds connection points, represented as dots, in a schematic to indicate wire connections and in a symbol file to indicate pin connections.

By default

- ❑ Crossed lines are not connected unless dotted.
- ❑ Wires joined at a T-intersection are connected, even without a dot.

Choose Tools – Options to change the default settings for dots.

Wire – Thick

Command

Changes selected wires to be displayed as thick.

Wire – Thin

Command

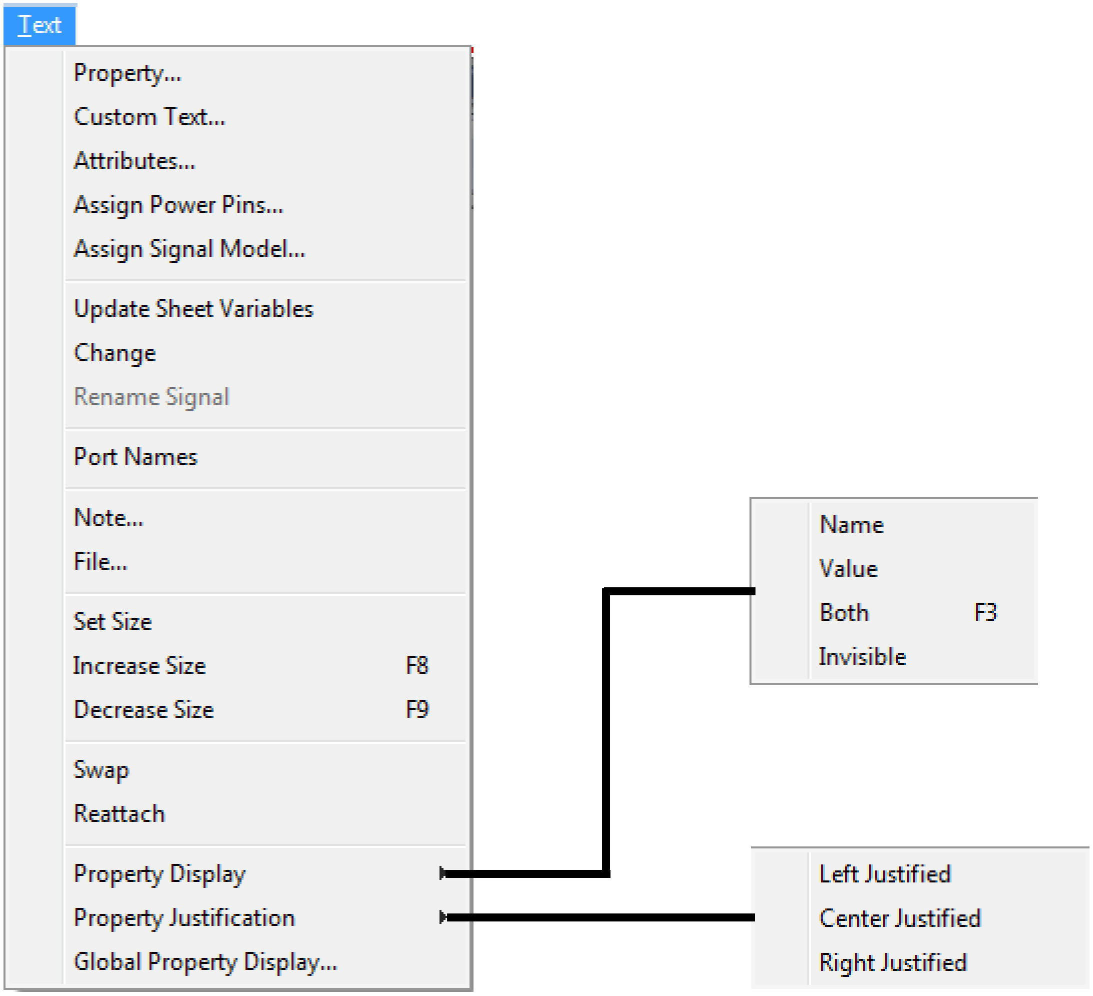
Changes selected wires to be displayed as thin.

Wire – Pattern

Command

Displays a dialog box from which you select a wire pattern.

Text Menu



Text – Property

Procedure Command

Displays a dialog box that lets you specify a property name and value.

Property name	Can be any string of text and underscores.
Property value	Can be any string of text and can include spaces and other punctuation.

When a property is added to a drawing, only the property value appears. Display – Properties temporarily displays both the names and values of properties on the drawing. Choose Tools – Options to change the default settings for properties.

Text – Custom Text

Procedure

Adds custom text to an object of the drawing. Use custom variables in the text to display context-specific information.

Place quotation marks around notes that begin with an open parenthesis. Quoted notes are not interpreted as commands.

Text – Attributes

Procedures Command

Displays the *Attributes* dialog box when you select an object on the drawing. Using the *Attributes* dialog box, you can work with many properties at once.

In occurrence editing mode, this dialog also displays any occurrence property information assigned to the selected object. When an occurrence property is added to a drawing, only the property value appears. Display – Properties temporarily displays both the names and values of properties on the drawing. Choose Tools – Options to change the default settings for properties.

Note: You cannot modify or delete section properties.

Procedures

- Adding Properties
- Displaying and Modifying Property Attributes
- Making an Attributes File

Text – Assign Power Pins

Procedure

Displays the *Assign Power Pins* dialog box that lets you view and edit properties on the power and ground pins of a component.

Text – Assign Signal Model

Displays the *SI Model Assignment* dialog box that lets you assign signal models to various devices.

Text – Update Sheet Variables

Procedure Command

Updates the custom text variables for page numbers on all pages in the design.

Text – Change

Procedure Command

Lets you modify selected notes, properties, and custom text. You can select multiple text items before editing.

Editing Properties and Notes

Use the arrow keys or the options in the pop-up menu to position the cursor in the text line. The options in the pop-up menu are as follows:

Option	Function
Done	Saves the changes and exits the change mode
Cancel	Brings the original text back by removing the changes that you had made
Next	Prompts you to click on the next text item in the group
Editor	Opens a text editor in which you can edit the text
Delete till End	Deletes the text from current position of the cursor till the end
Position at BOL	Positions the cursor at the beginning of the text
Position at EOL	Positions the cursor at the end of the text

You can also use the keyboard keys to move the cursor. Press *Del* or *Backspace* to delete a character or press CTRL+K to delete text from cursor to end of the line. Press *Enter* or *Return* to move to the next selection, or press the *Esc* button to cancel.

Note: While using the change command from the console command window, you cannot switch to another command in the console window. To use another command from the console window, press *Esc* or select *Done* in the pop-up menu, and type in the command.

Editing Custom Text

Custom Text does not appear along with other text. For editing the format of custom text, select the menu option *Text – Change* and click on the custom text. The *Custom Text* dialog box opens. You can change the *FORMAT string* of the custom text, and also add or remove the variables being used in the string. The *DISPLAY string* gets updated accordingly.

Text – Rename Signal

Lets you rename a signal.

Procedure Command

Text – Port Names

Procedure Command

Adds the port names from the symbol of the drawing.

The port names are added to an unused area of the schematic. This operation is used in hierarchical designs and for library development. It eliminates the need to retype signal names and reduces the chance of mislabeling signals.

Text – Note

Procedure Command

Adds text to a drawing. This text can also be a URL. Choose Tools – Options to change the default settings for text.

Text – File

Procedure Command

Imports text from a file. Each line in the file is converted to a note that can be individually moved, copied, deleted, or changed.

Text – Set Size

Procedure Command

Changes the size of the selected text.

Text – Increase Size

Procedure Command

Increases the size of the selected text incrementally.

Text – Decrease Size

Procedure Command

Decreases the size of the selected text incrementally.

Text – Swap

Procedure Command

Swaps two notes or properties. Only two properties or two notes can be swapped; notes and properties cannot be swapped with each other. Properties that you cannot swap are

- ❑ Hard properties
- ❑ Default properties
- ❑ Properties generated by

  Tools – Back Annotate

  Component – Section – Single Section

  Component – Swap Pins

Text – Reattach

Procedure Command

Reattaches a property, including signal names, from one object to another. For example, you can attach a property from the input pin of a device to the output pin.

Text – Property Display – Name

Procedure Command

Makes only property names visible on the drawing (without associated property values). Choose Tools – Options to change the default settings for property display.

Text – Property Display – Value

Procedure Command

Makes only property values visible on the drawing (without associated property names). Choose Tools – Options to change the default settings for property display.

Text – Property Display – Both

Procedure Command

Makes both property names and values visible on the drawing. Choose Tools – Options to change the default settings for property display.

Text – Property Display – Invisible

Procedure Command

Makes properties invisible on the drawing. Choose Tools – Options to change the default settings for property display.

Text – Property Justification – Left Justified

Adjusts the horizontal spacing so that property value (text) is aligned towards the left on the drawing. This is equivalent to the Align option of the Attributes dialog box.

Text – Property Justification – Center Justified

Adjusts the horizontal spacing so that property value (text) is aligned towards the center on the drawing.

Text – Property Justification – Right Justified

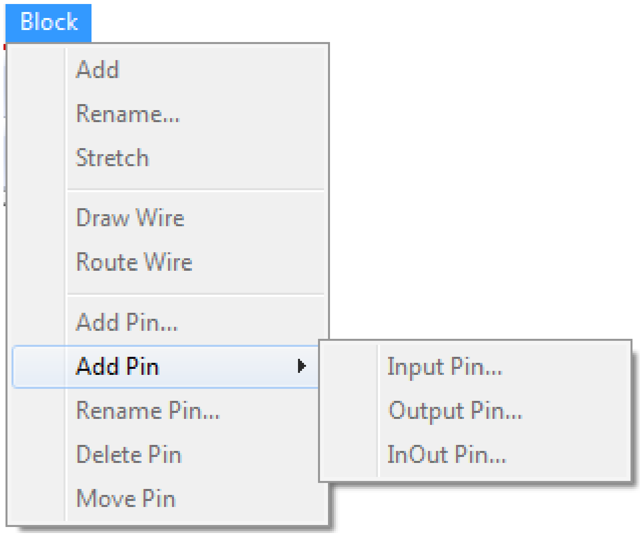
Adjusts the horizontal spacing so that property value (text) is aligned towards the right on the drawing.

Text – Global Property Display

Procedure

Displays the Global Property Visibility Change dialog box. Use this dialog box to change the name/value visibility of a property globally.

Block Menu



Block – Add

Procedure Command

Creates blocks in hierarchical designs. When you add a block, you can name it yourself or let Design Entry HDL name it. Design Entry HDL names the added block BLOCK*n*, where *n* is an integer—for example, BLOCK1.

Use the pop-up menu to name a block. If you specify an existing block name, a copy of the existing block attaches to the cursor for you to place on the drawing.

Note: Design Entry HDL will not let you add a block while zoomed in very far on a drawing. Zoom out before adding a block.

Block – Rename

Procedure Dialog Box Command

Renames a block. This is useful if

- Design Entry HDL has assigned names for added blocks, and you want to rename them.
- You want to create a block similar to an existing one—you can copy the block and then rename it.

If you have saved the schematic prior to renaming a block, Design Entry HDL does not remove the old block from disk. Choose File – Remove to remove the old block.

Block – Stretch

Procedure Command

Resizes a block. Pins adjust with the stretch of the block. Pins that point left or right move only horizontally. Pins that point up or down move only vertically.

Wires will be attached to the new pin locations but are not re-routed. Use Edit – Split and Edit – Delete to straighten any distorted wire connections as a result of stretching the block.

Block – Draw Wire

Procedure Command

Draws a wire that you manually route around objects to connect two blocks. (To automatically route a wire around objects, choose Block – Route Wire.)

If there are no block pins to connect the wire, Design Entry HDL adds the pins. Design Entry HDL names the pins PIN*n*, where *n* is an integer—for example, PIN1.

You can specify pin names (such as *data_addr*, *data_in*, or *data_out*) in the following three ways:

- ❑ Choose *Block – Draw Wire*. Draw the wire, click right and choose Signal Name to name the wire. If there are no block pins for the connection, Design Entry HDL adds the pins, naming them with the signal name that you can specify using the pop-up menu.
- ❑ Choose *Block – Add Pin* to add to add input, output or input pins before wiring a connection between them. For more information, see Block – Add Pin – Input Pin, Block – Add Pin – Output Pin and Block – Add Pin – InOut Pin.

- ❑ Let Design Entry HDL assign block pin names, and rename them using Block – Rename Pin.

Use the pop-up menu to do the following:

- ❑ Change the wire orientation (*Orientation*).
- ❑ Name the wire (*Signal Name*). Unnamed block pins assume the signal name that you specify. You can specify a bus name for the signal, such as *signal_name*[1..6].

Block – Route Wire

Procedure Command

Draws a wire to connect blocks, automatically routing it around objects in its path. The default wire orientation is orthogonal If Design Entry HDL cannot determine a route, a diagonal line is drawn directly between the blocks. (To manually route a wire around objects, choose Block – Draw Wire.)

If there are no block pins to connect the wire, Design Entry HDL adds the pins. Design Entry HDL names them PIN`n`, where n is an integer—for example, PIN1.

You can specify pin names (such as *data_addr*, *data_in*, or *data_out*) three ways:

- ❑ Choose *Block – Route Wire* and use the pop-up menu to name block pins. If there are no block pins for the connection, Design Entry HDL adds the pins, naming them with the signal name that you can specify using the pop-up menu.
- ❑ Choose *Block – Add Pin –* <`pin type`> to add pins before wiring a connection between them.
- ❑ Let Design Entry HDL assign block pin names, and rename them using Block – Rename Pin.

Use the pop-up menu to name the wire (*Signal Name*). Unnamed block pins assume the signal name that you specify. You can specify a bus name for the signal, such as `signal_name`[1..6].

Block – Add Pin

Command

Adds interface pins to blocks. This is useful if you want to define block pins before connecting blocks or if you already know what names you want to assign to pins. Alternatively, you can define interface pins on blocks before connecting blocks, using either *Block – Route Wire* or *Block – Draw Wire*.

Choose *Block – Add Pin* to specify pin names and place them on the block in the order you enter them.

Block – Add Pin – Input Pin

Procedure   Command

Adds input interface pins to blocks. This is useful if you want to define input block pins before connecting blocks or if you already know what names you want to assign to input pins.

Choose *Block – Add Pin – Input Pin* to specify pin names and place them on the block in the order you enter them. To toggle the pin mode before you place the pin on the block, click the right mouse button and choose Change Mode. Alternately, press *Ctrl* and click the left mouse button in a two-button mouse or click the middle mouse button in a three-button mouse.

Design Entry HDL adds the property `VHDL_MODE=IN` to the pin so as to retain the pin mode even if the pin location is changed.

The `VHDL_MODE=IN` property is used to declare an input pin. To manually add an input pin on a block, open the symbol view for the block, draw the pin on the block and add the `VHDL_MODE=IN` property on the pin using the *Attributes* dialog box. The `VHDL_MODE` property is not read by Design Entry HDL if added on the schematic.

Block – Add Pin – Output Pin

Procedure   Command

Adds output interface pins to blocks. This is useful if you want to define output block pins before connecting blocks or if you already know what names you want to assign to output pins.

Choose *Block – Add Pin – Output Pin* to specify pin names and place them on the block in the order you enter them. To toggle the pin mode before you place the pin on the block, click the right mouse button and choose Change Mode. Alternately, press *Ctrl* and click the left mouse button in a two-button mouse or click the middle mouse button in a three-button mouse.

Design Entry HDL adds the property `VHDL_MODE=OUT` to the pin so as to retain the pin mode even if the pin location is changed.

The `VHDL_MODE=OUT` property is used to declare an output pin. To manually add an output pin on a block, open the symbol view for the block, draw the pin on the block and add the `VHDL_MODE=OUT` property on the pin using the *Attributes* dialog box. The `VHDL_MODE` property is not read by Design Entry HDL if added on the schematic.

Block – Add Pin – InOut Pin

Procedure Command

Adds inout interface pins to blocks. This is useful if you want to define inout block pins before connecting blocks or if you already know what names you want to assign to inout pins.

Choose *Block – Add Pin – InOut Pin* to specify pin names and place them on the block in the order you enter them. To toggle the pin mode before you place the pin on the block, click the right mouse button and choose Change Mode. Alternately, press *Ctrl* and click the left mouse button in a two-button mouse or click the middle mouse button in a three-button mouse.

Design Entry HDL adds the property `VHDL_MODE=INOUT` to the pin so as to retain the pin mode even if the pin location is changed.

The `VHDL_MODE=INOUT` property is used to declare an inout pin. To manually add an inout pin on a block, open the symbol view for the block, draw the pin on the block and add the `VHDL_MODE=INOUT` property on the pin using the *Attributes* dialog box. The `VHDL_MODE` property is not read by Design Entry HDL if added on the schematic.

Block – Rename Pin

Procedure Dialog Box Command

Renames a block pin.

Block – Delete Pin

Procedure Command

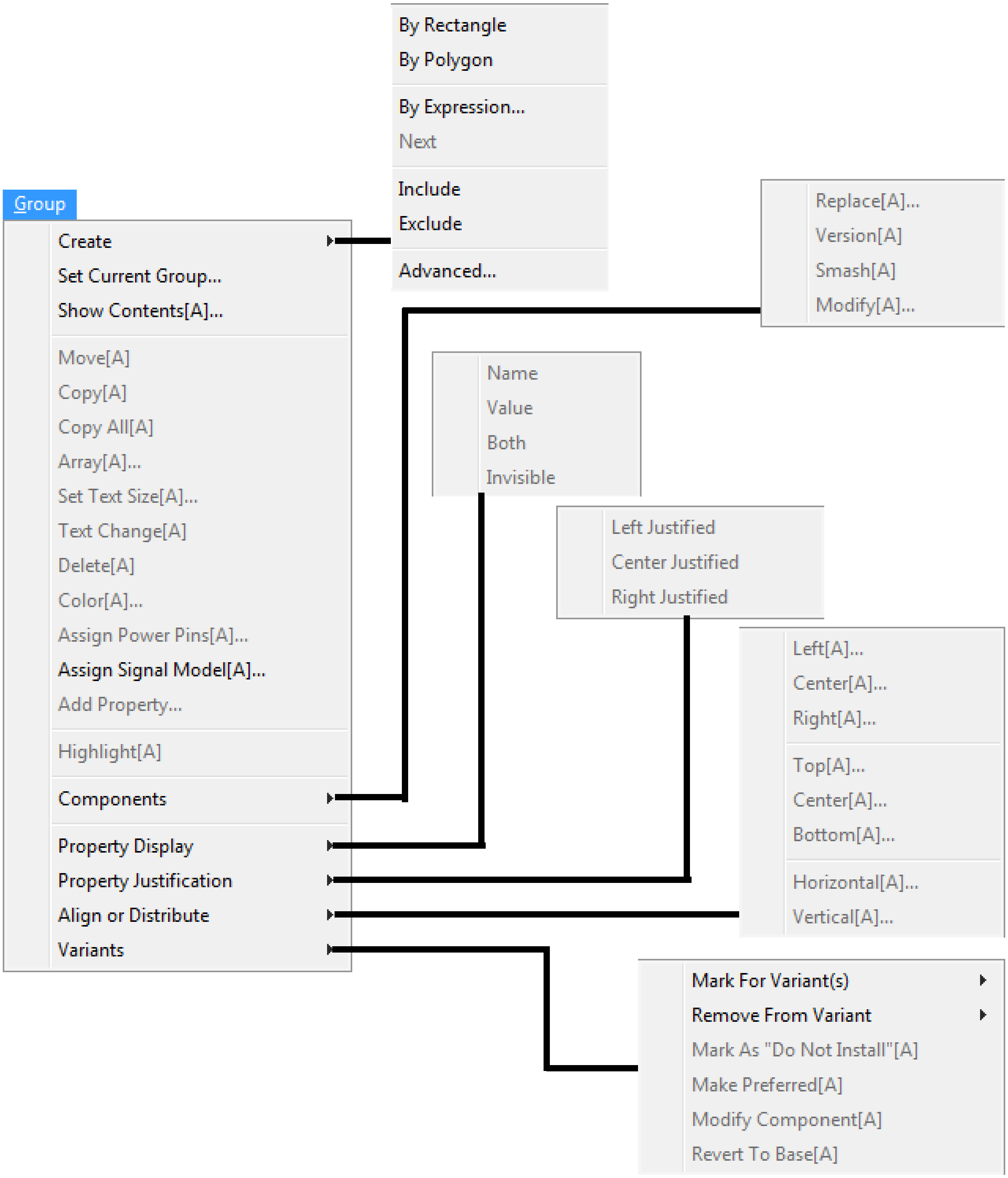
Deletes a block pin.

Block – Move Pin

Procedure Command

Moves a pin from one location on a block to a new location on the same block. You cannot move pins across components.

Group Menu


Group
Create
Set Current Group...
Show Contents[A]...
Move[A]
Copy[A]
Copy All[A]
Array[A]...
Set Text Size[A]...
Text Change[A]
Delete[A]
Color[A]...
Assign Power Pins[A]...
Assign Signal Model[A]...
Add Property...
Highlight[A]
Components
Property Display
Property Justification
Align or Distribute
Variants
By Rectangle
By Polygon
By Expression...
Next
Include
Exclude
Advanced...
Replace[A]...
Version[A]
Smash[A]
Modify[A]...
Name
Value
Both
Invisible
Left Justified
Center Justified
Right Justified
Left[A]...
Center[A]...
Right[A]...
Top[A]...
Center[A]...
Bottom[A]...
Horizontal[A]...
Vertical[A]...
Mark For Variant(s)
Remove From Variant
Mark As "Do Not Install"[A]
Make Preferred[A]
Modify Component[A]
Revert To Base[A]

Group – Create – By Rectangle

Procedure Command

Groups objects within a stretchable rectangle. Grouped objects appear highlighted. You can group components, wire segments, dots, properties, and notes. Groups are named with a single-letter identifier. Design Entry HDL automatically uses the current group letter. You can specify the group name using Group – Set Current Group.

Use the pop-up menu to group the entire schematic (All).

To view the contents of a group, choose Group – Show Contents [A]. To modify a group to include additional objects or exclude certain objects, choose Group – Create – Include or Group – Create – Exclude.

Group – Create – By Polygon

Procedure Command

Groups objects within a polygon. Grouped objects appear highlighted. You can group components, wire segments, dots, properties, and notes. Groups are named with a single-letter identifier. Design Entry HDL automatically uses the current group letter. You can specify the group name using Group – Set Current Group.

Use the pop-up menu to

- ❑ Group the entire schematic (*All*)
- ❑ Close the polygon (*Next*)

To view the contents of a group, choose Group – Show Contents [A]. To modify a group to include additional objects or exclude certain objects, choose Group – Create – Include or Group – Create – Exclude.

Group – Create – By Expression

Procedure Command

Finds objects matching the specified pattern string and groups them together. The pattern can be used to match component names, notes, property names, property values, or signal names. Properties can also be searched for by specifying both name and value separated by an equal sign.

Wildcards are allowed in the pattern. An asterisk matches any number of characters and a question mark matches any single character. The pattern is not case sensitive.

You can add the objects grouped using *Group – Create – By Expression*, to an already existing group. To do this, first choose Group – Set Current Group and then *Group – Create – By Expression*.

To view the contents of a group, choose Group – Show Contents [A]. To modify a group to include additional objects or exclude certain objects, choose Group – Create – Include or Group – Create – Exclude.

Note: This feature is available only for single page schematics.

Group – Create – Next

Command

Is used to traverse the objects in a group created by Group – Create – By Expression. *Group – Create – Next* highlights an object in the group by drawing a blinking rectangle around it. After performing the desired operation on the selected object, you can proceed to the next item by selecting *Group – Create – Next* again. You can only step through the list of located objects once.

Group – Create – Include

Procedure Command

Adds items or groups to the current group.

Group – Create – Exclude

Procedure Command

Removes items or groups from the current group.

Group – Create – Advanced

Displays the Group Controls dialog box. The Group Controls dialog box provides a single place from where you can perform various operations on a group of objects.

Group – Set Current Group

Procedure

Lets you set the letter that identifies the current group. The selected group name is applied to all the commands in the group menu. This means that all future commands, such as group create, edit, next, and highlight operations, are performed within the selected group.

Group – Show Contents [*A*]

Procedure

Displays the *Group Contents* dialog box, where you can

- ❑ View the contents of groups you have defined in the current session.
- ❑ Turn on and off highlighting for a group.

Group – Move [*A*]

Procedure Command

Moves the current group from one location to another in the current drawing or between drawings in different windows. Choose Edit – Move to move individual objects.

Design Entry HDL preserves the electrical connectivity of objects with wires attached.

Use the pop-up menu to automatically change attachments for a group you are moving to another location in the design.

Group – Copy [*A*]

Procedure Command

Copies the current group (excluding properties) in the current drawing or between drawings in different windows. You can also copy individual objects and certain properties by choosing Edit – Copy.

Note: You cannot copy groups of properties.

Use the pop-up menu to:

- ❑ Copy All properties (section properties, soft properties, pin properties, wire properties, and properties attached to each other) with the group. This is useful if you are copying a section of logic from one drawing to another. You can also choose Group – Copy All [A] to copy properties with a group of objects.
- ❑ Retain the specified group and copy it in several unrelated places on the drawing without having to re-select the group (*Retain*/*Terminate Selection*).

Choose Group – Array [A] to create multiple copies of groups.

Group – Copy All [*A*]

Procedure Command

Copies a group, including properties, in the current drawing or between drawings in different windows. This is especially useful if you are copying a section of logic from one drawing to another. You can also copy individual objects with their properties by choosing Edit – Copy All.

Use the pop-up menu to retain the specified group and copy it in several unrelated places on the drawing without having to re-select the group (*Retain*/*Terminate Selection*).

Choose Group – Copy [A] to copy a group, excluding properties, or choose Group – Array [A] to create multiple copies of groups.

Group – Array [*A*]

Procedure Command

Makes multiple copies of a group in the current drawing or between drawings in different windows. You cannot copy groups of properties. Properties attached to objects are copied with the group. You can also make multiple copies of individual objects by choosing Edit – Array.

- ❑ *Copy All* properties (section properties, soft properties, pin properties, wire properties, and properties attached to each other) with the group. This is useful if you are copying a section of logic from one drawing to another. You can also choose Group – Copy All [A] to copy properties with a group of objects.
- ❑ Retain the specified group and copy it in several unrelated places on the drawing without having to re-select the group (*Retain*/*Terminate Selection*).

Choose Group – Copy [A] to copy a group, excluding properties, or choose Group – Copy All [A] to copy a group of objects, including properties.

Group – Set Text Size[A]

Procedure Command

Changes the size of text in a group.

Group – Text Change [*A*]

Procedure Command

Selects all the text items in the specified group and lets you begin editing text.

Note: When changing text on grouped objects, it is easiest to use the Text Change Editor (Ctrl+E).

Group – Delete [*A*]

Procedure Command

Deletes all objects in the current group. To delete individual objects, choose *Edit – Delete*.

You cannot delete default properties on components or pin number properties on swapped pins.

Edit – Undo lets you cancel a deletion.

Group – Color [*A*]

Procedure Command

Displays the color toolbar from which you select colors for the current group.

Choose Edit – Color to select colors for individual objects. Choose Tools – Options to establish default colors for objects and window background.

On a monochrome display, use Display – Color to view the current color of objects.

Group – Assign Power Pins [A]

Procedure

Displays the *Assign Power Pins* dialog box that lets you view and edit properties on the power and ground pins of a group of components.

Group – Assign Signal Model [A]

Procedure

Displays the *SI Model Assignment* dialog box that lets you assign a signal model to a group. Choose this menu option and then select a group.

Group – Add Property

Command

Opens the *Add Property window* to add user properties to groups of objects. This is equivalent to the `auto property` command.

Group – Highlight [*A*]

Procedure Command

Highlights objects in the current group.

Group – Components – Replace [*A*]

Procedure Command

Displays the *Replace Component* dialog box. Select the component that should replace all components in the current group. All the components in the current group are replaced with version 1 of the component that you selected in the *Replace Component* dialog box.

When you replace components

- ❑ Property names and default values are retained, except those generated from

  Tools – Back Annotate

Component – Section – Single Section

Component – Swap Pins

- ❑ Default properties with a value of ? retain this value.
- ❑ Wire connections are retained if the pins are in the same location as on the original components.
- ❑ Unnamed wires are deleted.
- ❑ Rotation is preserved.
- ❑ Pin properties are reattached if pin names are the same as on the original components. Unmatched pin names cause pin properties to become symbol properties.

To replace a single component, choose Component – Replace.

Group – Components – Version [*A*]

Procedure Command

Replaces components defined as part of a group with their next version. Design Entry HDL determines the current version of a component and displays the next version in sequence.

You can also:

- Replace a single component using Component – Replace.
- Rotate (Edit – Rotate), spin (Edit – Spin), or mirror (Edit – Mirror) a component.

Group – Components – Smash [*A*]

Procedure Command

Breaks up all components in the current group into their individual elements. Any properties attached to components are deleted. This is useful for creating library component drawings.

Choose Component – Smash to break up individual components.

Group – Components – Modify [*A*]

Procedure Command

Displays list of physical packages associated with the selected part. The *Group – Components – Modify* menu option works on components with the same part name. If all the components have the same part name, Design Entry HDL opens the *Physical Part Filter* with all the part rows. Select a new row in the filter and click *OK*. This will replace existing components with the new selection along with the key properties.

Group – Property Display – Name

Procedure

Makes property names visible for all objects in the current group on your drawing without associated property values. You can change the default settings for properties in Tools – Options.

Group – Property Display – Value

Procedure

Makes property values visible for all objects in the current group on your drawing without associated property names. You can change the default settings for properties in Tools – Options.

Group – Property Display – Both

Procedure

Makes both property names and values visible for all objects in the current group on your drawing. You can change the default settings for properties in Tools – Options.

Group – Property Display – Invisible

Procedure

Makes properties invisible for all objects in the current group on your drawing. You can change the default settings for properties in Tools – Options.

Group – Property Justification – Left Justified

Adjusts the horizontal spacing so that the visible property values (text) of all the objects in the group are aligned towards the left on the drawing.

Group – Property Justification – Center Justified

Adjusts the horizontal spacing so that the visible property values (text) of all the objects in the group are aligned towards the center on the drawing.

Group – Property Justification – Right Justified

Adjusts the horizontal spacing so that the visible property values (text) of all the objects in the group are aligned towards the right on the drawing.

Group – Align or Distribute

Align or distributes a group of objects, either vertically or horizontally. The Align function aligns a selected set of objects with respect to a common axis. The Distribute function equally spaces a group of objects according to the type of distribution—horizontal or vertical.

- Align Left
- Align Center
- Align Right
- Align Top
- Align Middle
- Align Bottom

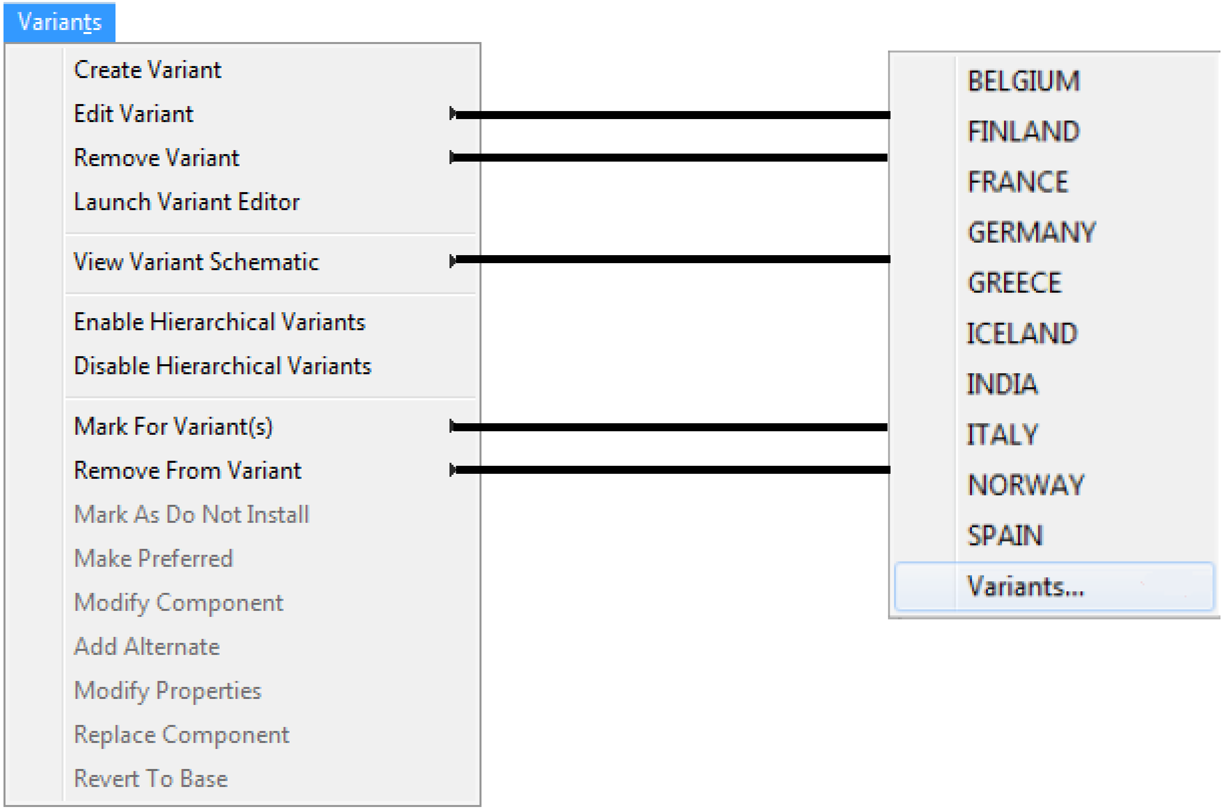
- Distribute Horizontally
- Distribute Vertically

See Alignment and Distribution in *Allegro Design Entry HDL User Guide* for more information.

Group – Variants

Allows you to group components and run variant-related commands for the group.

Variants Menu



When you open a design in Design Entry HDL, the design is displayed in the base schematic view. The Variants menu provides commands that enable you to manage variants in your design. These commands let you create, edit, and remove variants. You can also select multiple objects on the schematic and mark them for modification in variants.

The menus also provide you the option to view variants on the schematic. When you choose a variant to view in the schematic, variant-specific data is displayed in the schematic and additional menu commands are enabled that allow you to perform variant-specific operations.

Variants – Create Variant

Allows you to create a variant, a variation of the base design created to generate a separate product.

Procedure

Variants – Edit Variant

Allows you to edit an existing variant.

Procedure

Variants – Remove Variant

You can remove a variant from the variant database using this option. To remove multiple variants at a time, choose *Variants — Remove Variant — Variants*.

Procedure

Variants – Launch Variant Editor

Use this option to launch Variant Editor from the schematic. Certain advanced operations such as creating functions and groups, defining components as alternates, marking an alternate as a preferred component, generating BOM reports, or replacing components can only be done in Variant Editor.

On saving changes in Variant Editor, the schematic is automatically synchronized and variant-specific changes can be viewed in the schematic sheet.

Procedure

Variants – View Variant Schematic

To view changes made for a particular variant, you can open a variant schematic view. You can switch from the base schematic view to any variant view using *Variants — View Variant Schematic — `<variant name>`*.

When you switch to the variant view, the title bar of the schematic window indicates that you are now in the viewer mode. This is a viewer mode only and schematic changes made in this mode cannot be saved. You can however make changes, plot, or publish the modified schematic.

Procedure

Variants – Enable Hierarchical Variants

Use this option to enable hierarchical variants for a block. Variants defined in lower-level hierarchical blocks can be applied on block instances in higher-level blocks. These variants are referred to as hierarchical variants in DE-HDL documentation.

All reusable blocks that contain variant definitions can be used for specifying block-level variants.

Procedure

Variants – Disable Hierarchical Variants

If you have enabled hierarchical variants for a particular block, you can choose to disable it for that block. When you disable hierarchical variants for a block, hierarchical variants for the lower-level reusable block are unloaded and the block reverts to the base state.

To disable the hierarchical variant option for a block, select a block on the schematic then use this option.

Procedure

Variants – Mark for Variant(s)

Use this option to identify components for a variant and change or customize their values. You can mark components for a single variant or for multiple variants at a time. You can also select multiple components and mark them for one variant or multiple variants.

Procedure

Variants – Remove from Variant

You can remove one or more components from selected variants. To do so, first ensure that you are in the required variant view.

Procedure

Variants – Mark as Do Not Install

In the variant schematic view, you can mark one or more components as Do Not Install (DNI) components.

Procedure

Variants – Make Preferred

You can mark one or more components as preferred components in a design. If the selected component was previously marked as DNI or was already a preferred component with a different value, the selected component reverts to the base schematic value.

The component also reverts to the display settings defined for the base schematic value (for details, see the Variant Overlay Options section of the Design Variance User Guide).

Procedure

Variants – Modify Component

You can modify components by changing their attribute values. To modify components, ensure that you are in the required variant view.

Procedure

Variants – Add Alternate

To add an alternate for a component in the schematic canvas, ensure that you are in the variant schematic view.

Procedure

Variants – Modify Properties

You can add or modify the user-defined properties of a component for a specific variant.

Procedure

Variants – Replace Component

Allows you to replace a component in a function or a variant with another component, if needed.

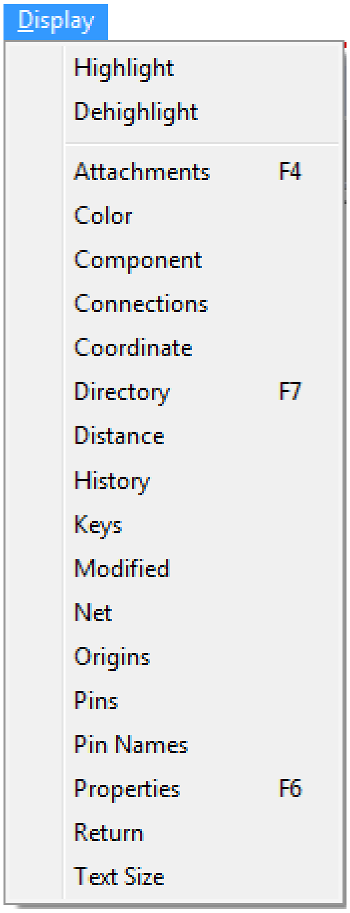
After defining functions and variants, Variant Editor allows you to replace a component in a function or a variant with another component, if needed. If a component is part of an alternate group, it cannot be replaced. The replaced component can have a different name or properties, but it should have the same or compatible footprint (JEDEC_TYPE property value).

Procedure

Variants – Revert to Base

Use this option to revert to the base schematic.

Display Menu



Display – Highlight

Procedure Command

Highlights selected objects in drawings, between drawings, and between Design Entry HDL and other system tools. For example, you can also correlate wires (nets), components, and pins in Design Entry HDL with the same objects in your layout tool.

Choose Display – Dehighlight to remove highlighting.

Display – Dehighlight

Procedure Command

Dehighlights pins, wires (nets), and components that have been highlighted using Display – Highlight.

Display – Attachments

Procedure Command

Displays attachments between visible properties and associated objects.

Display – Color

Procedure Command

Displays the color of a selected object.

The color is displayed in the Design Entry HDL console window and as ticker text in the status bar.

Display – Component

Procedure

Displays component information (library, version, and angle).

Display – Connections

Procedure

Displays an asterisk at each wire connection on the drawing (if a connection is not already indicated by a dot).

Display – Coordinate

Procedure Command

Displays the x, y location of a selected point in the drawing.

Where x and y can range from +18500 to -18500.

Display – Directory

Procedure Command

Displays the current project directory.

The current directory is displayed in the Design Entry HDL console window and as ticker text in the status bar.

Display – Distance

Procedure Command

Displays the distance between two selected points in the drawing.

- Clicking the left mouse button at two grid points tells you the distance between the two grid points.

- Clicking the middle mouse button (in a three button mouse) or Ctrl+left mouse button (in a two button mouse) at two points on the screen displays the distance between the two screen points.

The distance is displayed in the Design Entry HDL console window and as ticker text in the status bar.

Display – History

Procedure Command

Lists all the drawings read in during the current session.

Display – Keys

Procedure Command

Lists function keys and their command assignments. Choose *Tools – Customize* to change function key settings.

Display – Modified

Procedure Command

Lists the drawings that were modified but not saved during the current session.

Display – Net

Procedure Command

Highlights a selected wire (net) and displays the net name.

The net name is displayed in the Design Entry HDL console window and as ticker text in the status bar.

Display – Origins

Procedure Command

Displays an asterisk at each object origin.

Display – Pins

Procedure Command

Displays an asterisk at each pin location.

Display – Pin Names

Procedure

Displays pin names on a selected component.

Display – Properties

Procedure Command

Displays the names and values of all properties, including any invisible properties, on the drawing.

Display – Return

Procedure Command

Displays the names of the previously viewed drawings in the drawing hierarchy, in order of return. These are the drawings you'll return to if you choose *File – Return*.

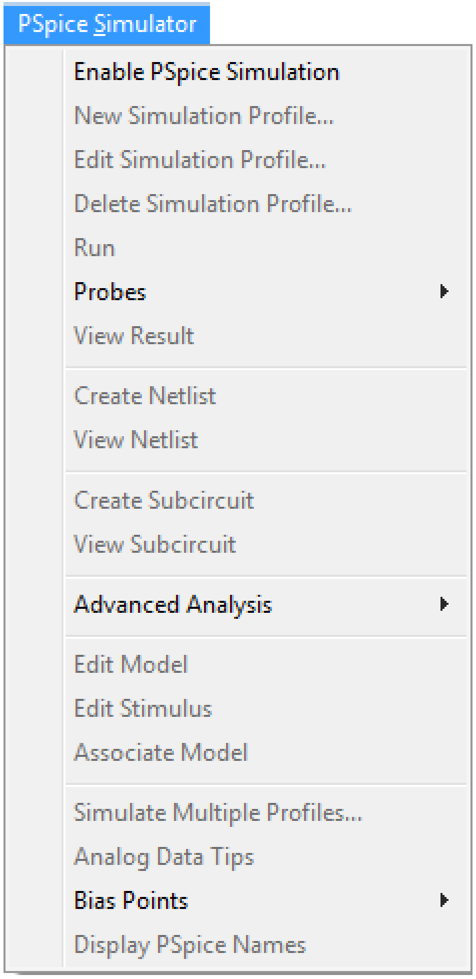
Display – Text Size

Procedure Command

Displays the size of the selected text (property name, property value, signal name, note, URL) in inches. This is the text size you specify in the Text page of the Design Entry HDL Options dialog box.

The text size is displayed in the Design Entry HDL console window and as ticker text in the status bar.

PSpice Simulator Menu



PSpice Simulator– Enable PSpice Simulation

Allows you to enable the PSpice Simulator menu items. The PSpice Simulator menu items are not enabled by default in Design Entry HDL. To enable the menu items, choose *PSpice Simulator – Enable PSpice Simulation*. After this, the design is enabled for PSpice simulation and the *PSpice – Enable PSpice Simulation* menu item is disabled and cannot be enabled again for the project.

For more information on simulation profiles, see the *PSpice A/D documentation*.

PSpice Simulator – New Simulation Profile

Allows you to create a PSpice simulation profile. Simulation profiles are used to save the simulation settings for an analysis type so that you can reuse the settings easily.

For more information on simulation profiles, see the *PSpice A/D documentation*.

PSpice Simulator – Edit Simulation Profile

Allows you to modify the simulation settings of the current simulation profile.

For more information on simulation profiles, see the *PSpice A/D documentation*.

PSpice Simulator – Delete Simulation Profile

Allows you to delete simulation profiles.

For more information on simulation profiles, see the *PSpice A/D documentation*.

PSpice Simulator – Run

Run simulation using the PSpice Simulator. The netlist is created and the simulation results are displayed in the PSpice Simulator Probe window.

PSpice Simulator – Probes – View Probes

Allows you to place Plot Window Templates in your design and also displays a list of probes already added to the design. Probes are the components that are placed on a schematic to indicate the points for which you want to see simulation waveforms displayed in PSpice Simulator. You can place probes on a part, net or pin.

To add a Plot Window Template

1. Click the *Templates* button.
2. From the displayed list select the required Plot Window Template.
3. Click the Place button.

The selected Plot Window Template gets added to the list of probes to be displayed and a template probe is attached to the cursor.

4. Place the template probe on the appropriate net, part or pin in the schematic for which you want to view the Plot Window Template.

   The check box next to the Plot Window Template should get selected.

To change the color assigned to a probe

1. Click on the color of the probe to display a drop-down list of colors.
2. Select the color you want to assign to the probe.

To view the name of a probe in the Design Entry HDL or PSpice format

You can view the name of the probe in the PSpice format or the Design Entry HDL hierarchical name format.

- Select the *PSpice* option to display the name of the probe in the PSpice format. If you move the mouse pointer over a probe in the list, you can view the name of the probe in the Design Entry HDL hierarchical name format.
- Select the *Design Entry* option to display the name of the probe in the Design Entry HDL hierarchical name format. If you move the mouse pointer over a probe in the list, you can view the name of the probe in the PSpice format.

To navigate to the location of probe in the schematic

1. Select the *Navigate* check box.
2. Double-click on a probe in the list. The location of the probe is highlighted in the schematic.

For more information on probes, see the *PSpice A/D documentation*.

PSpice Simulator – Probes – Voltage Probe

Allows you to place a voltage probe on a net.

1. From the Probes submenu, select Voltage Probe.
2. Click on the net on which you want to place a probe.

The probe is displayed in the list with a color assigned to the probe. The color you have specified for the probe will be reflected in the color of the waveform for the probe in PSpice Simulator.

PSpice Simulator – Probes – Current Probe

1. From the Probes submenu, select Current Probe.
2. In the schematic, click on the pin on which you want to place a probe.

   The probe is displayed in the list with a color assigned to the probe. The color you have specified for the probe will be reflected in the color of the waveform for the probe in PSpice Simulator.

PSpice Simulator – Probes – Power Probe

1. From the Probes submenu, select Power Probe.
2. In the schematic, click on the part or the device on which you want to place the probe.

   The probe is displayed in the list with a color assigned to the probe. The color you have specified for the probe will be reflected in the color of the waveform for the probe in PSpice Simulator.

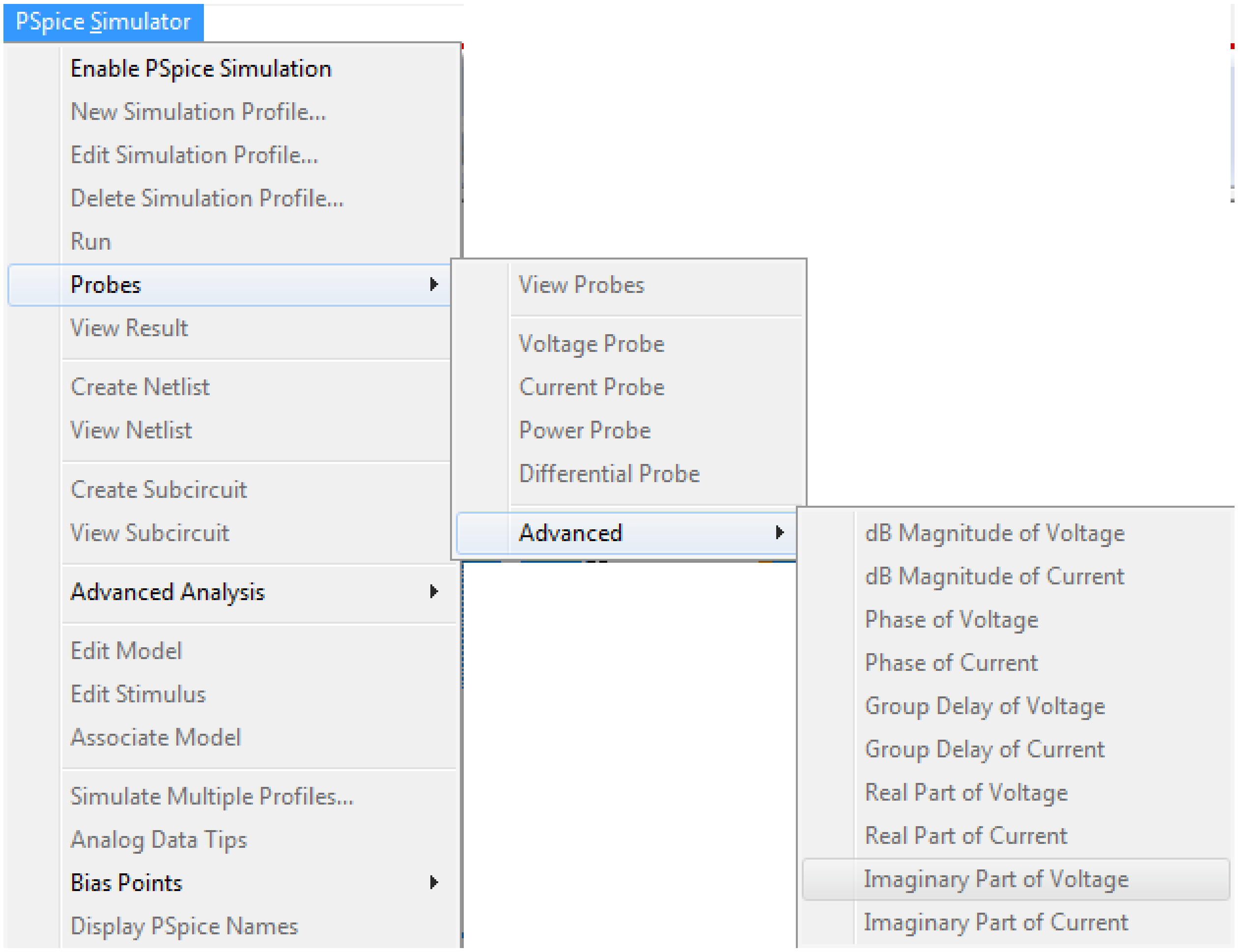
PSpice Simulator – Probes – Differential Probe

Allows you to put a differential probe on a net.

To place a differential probe:

1. Choose *PSpice Simulator – Probes – Differential Probe*.
2. In the schematic, click the net on which you want to place the probe.

PSpice Simulator – Probes – Advanced



Use these commands to place markers for AC Sweep/Noise analysis.

Note: Marker types on the Advanced command submenu are only available after defining a simulation profile for an AC Sweep/Noise analysis.

PSpice Simulator – View Result

Allows you to view the simulation results in the PSpice Simulator A/D Probe window.

PSpice Simulator – Create Netlist

Allows you to create the PSpice Simulator netlist for the design. The netlist contains a list of device names, values, and how they are connected with other devices.

PSpice Simulator – View Netlist

Allows you to view the PSpice Simulator netlist in a text editor.

PSpice Simulator – Create Subcircuit

Allows you to create the `.SUBCKT` for the design. The generated `SUBCKT` is saved in `<designname>_subckt.lib` in the `psp_sim_1` view.

PSpice Simulator – View Subcircuit

Allows you to view the PSpice Simulator subckt in a text editor.

PSpice Simulator – Advanced Analysis – Sensitivity

Displays the Sensitivity tool that allows you to examine how much each component affects circuit behavior by itself and in comparison to the other components. It also varies all tolerances to create worst-case (minimum and maximum) measurement values.

You can use Sensitivity to identify the sensitive components, then export the components to Optimizer to fine-tune the circuit behavior.

You can also use Sensitivity to identify which components affect yield the most, then tighten tolerances of sensitive components and loosen tolerances of non-sensitive components. With this information you can evaluate yield versus cost trade-offs.

For more information on the Sensitivity tool, see *PSpice Simulator Advanced Analysis User's Guide.*

PSpice Simulator – Advanced Analysis – Optimizer

Displays the Optimizer tool that allows you to analyze analog circuits and systems. It helps you modify and optimize analog designs to meet your performance goals.

Optimizer fine-tunes your designs faster than trial and error bench testing can. Use Optimizer to find the best component or system values for your specifications.

For more information on the Optimizer tool, see *PSpice Simulator Advanced Analysis User's Guide.*

PSpice Simulator – Advanced Analysis – Monte Carlo

Displays the Monte Carlo tool that allows you to predict the statistical behavior of a circuit when part values are varied within tolerance. Monte Carlo also calculates yield, which can be used for mass manufacturing predictions.

For more information on the Monte Carlo tool, see *PSpice Simulator Advanced Analysis User's Guide*.

PSpice Simulator – Advanced Analysis – Smoke

Displays the Smoke tool that allows you to evaluate component stress due to power dissipation, increase in junction temperature, secondary breakdowns, or violations of voltage / current limits.

For more information on the Smoke tool, see *PSpice Simulator Advanced Analysis User's Guide*.

PSpice Simulator – Advanced Analysis – Parametric Plotter

Allows you to open the Parametric Plotter window. You can use the Parametric Plotter to perform sweep analysis. Using the Parametric Plotter, you can do the following:

- Sweep multiple parameters.
- Allow device/model parameters to be swept.
- Display sweep results in spreadsheet format.

- Plot measurement results in the Probe dialog.
- Post analysis measurement evaluation.

Note: Parametric Plotter is available only if you have the SPice Simulator Advanced Analysis license.

PSpice Simulator – Edit Model

Allows you to invoke the PSpice Model Editor tool to edit a model definition for a part instance on your schematic. For more information on the Model Editor tool, see the *Model Editor online help* and the see the *PSpice Simulator Advanced Analysis User's Guide*.

PSpice Simulator – Edit Stimulus

Allows you to invoke the PSpice Stimulus Editor tool to setup the stimulus for the selected part in the schematic. For more information on the Stimulus Editor tool, see the *Stimulus Editor online help* and the *PSpice Simulator documentation*.

PSpice Simulator – Associate Model

This command is available only if you have a symbol open in the symbol view. Invokes Model Import wizard for associating a PSpice model to the open symbol.

To view the help for Model Import Wizard, see Model Import Wizard (Select Matching) and Model Import Wizard (Define Pin Mapping).

PSpice Simulator – Simulate Multiple Profiles

Allows you to select multiple profiles you want to simulate in one run.

PSpice Simulator – Analog Data Tips

Allows you to view the netlist representation of a part or net. Select this menu option and move the mouse pointer over a part. The representation of the part in the PSpice netlist is displayed.

For example, suppose that a resistor is instantiated as I5 in the schematic and is connected to the net MID.

1. Move the mouse pointer over the resistor. Design Entry HDL displays "Instance I5 of R".
2. From the *PSpice Simulator* menu choose *Analog Data Tips*.
3. Move the mouse pointer over the resistor. Design Entry HDL displays "Instance R4 of R" which is the netlist representation of the resistor according to the sample netlist given below:

```
V1      N001 0   STIMULUS=SINE
V2      in N001 0Vdc
C1      mid out  0.47U
R1      0 out  5.6K
R2      0 mid  3.3K
R3      mid ad1  3.3K
R4      in mid  1K
D1      0 mid D1N3940
D2      mid ad1 D1N3940
V3      ad1 0 5VDC
```

PSpice Simulator – Bias Points – Enable

Procedure

Enables displaying bias point values on the schematic. If this option is not selected, none of the bias points values will be displayed on the screen.

PSpice Simulator– Bias Points – Annotate Bias Values

Procedure

Updates the schematic with latest bias point values.

PSpice Simulator – Bias Points – Enable Bias Voltage Display

Procedure

Displays bias point voltages on all the nodes in a schematic. Bias voltages are displayed next to their corresponding nodes.

PSpice Simulator – Bias Points – Enable Bias Current Display

Procedure

Displays bias point currents on all modeled pins throughout the design. Bias currents are displayed next to their corresponding device pins. For currents on pins, a positive value for the bias point means current is flowing into the pin, while a negative value indicates the current flowing out of the pin.

PSpice Simulator – Bias Points – Enable Bias Power Display

Procedure

Displays bias point power dissipation of the device.

PSpice Simulator – Bias Points – Preferences

Dialog box Procedure

Displays the Bias Point Preferences dialog box, that can be used to change the default color, font, and precision value for the bias point values.

PSpice Simulator – Display PSpice Names

Controls the display of the `$PSPICE_LOCATION` value on the schematic. Choose this menu option to display the `$PSPICE_LOCATION` value. If this option is not selected, the `$LOCATION` value is displayed provided the Attributes form specifies the value to be shown.

PSpice Simulator – Design Name

This option is available in the Analog toolbar. It is a drop-down field.

Displays the name of the current design. Click on the drop-down to select another design that you want to simulate using PSpice Simulator.

The names of all the designs in the project are displayed in this drop-down list.

Spice Simulator – Profile Name

This option is available in the Analog toolbar. It is a drop-down field.

Displays the name of the currently loaded PSpice Simulator simulation profile. Click on the drop-down to select another PSpice Simulator simulation profile.

All the simulation profiles for the design are displayed in this drop-down list.

For more information on simulation profiles, see the PSpice Simulator documentation.

RF-PCB Menu

This menu is visible if you select the *Enable Pre-select Mode* option in Design Entry HDL – Tools – Options.

RF-PCB – Import IFF

Procedure

Starts the RF-PCB IFF Import wizard, which helps you import a schematic IFF file into a project.

RF-PCB – RF Group – Add Group

Procedure

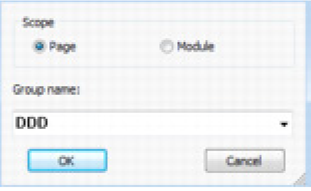
The Add Group option enables you to attach a property (RFGROUP) to the selected components.

RF-PCB – RF Group – Add Split

Select a wire or multiple wires and then use this option. The RFSPLIT property is attached to the wires selected. You can only select wires in the current page for this command. You cannot use this option after selecting wires that cross pages.

RF-PCB – RF Group – Disband

Use this option to ungroup poperties. When you select this option, the Disband Group dialog box is displayed. Choose a group from the *Group name* drop-down list then select the scope, that is, Page, or Module, and click *OK*.



Scope
● Page ○ Module
Group name: DDD

The RFGROUP property will be removed from each component of the selected group.

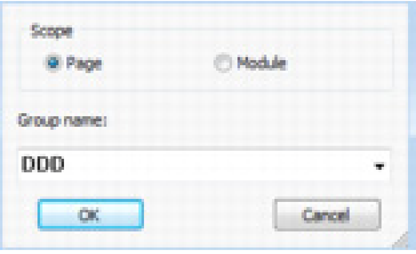
RF-PCB – RF Group – Exclude

Select one or more components with the RFGROUP property attached or one or more wires with RFSPLIT attached then use this option.

The property is removed for the selected objects. This command only works for the current page objects.

RF-PCB – RF Group – Display Group

Select a group or the All option from the drop-down list to display one or all groups. To display a group that includes elements from other pages, select the Module radio button.

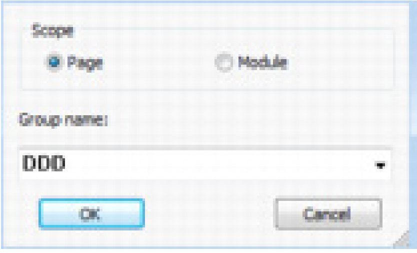


Scope
● Page ○ Module
Group name: DDD

When you click OK, all the components in the selected groups are listed in the command line. If the Module radio button is selected, the components of the selected groups are highlighted in the current page.

RF-PCB – RF Group – Display Split

When you use thisoption, the following dialog box is displayed:


Scope

● Page ○ Module

Group name:

DDD

When you click *OK*, wires with the RFSPLIT property are highlighted and a description of each wire with the RFSPLIT property is displayed in the command line.

For a description of each wire with the RFSPLIT property in the current page, select the Page option. Select the Module radio button for a description of each wire with the RFSPLIT property in the complete design.

Design Management Menu

This menu option is enabled or grayed out depending on the licenses available to you. Using the Design Management menu option, you can enable structured team design for a design to shorten your design cycle. You can manage the design on a file system, in a folder on SharePoint, or in a folder in PTC Windchill.

You can also manage the changes to a design using this option. Managing changes to a design is referred to as data management in the Design Entry HDL documentation.

For details about these menu options, see *Allegro Design Management User Guide*.

Design Management – Enable Design Management

Use this option to enable a design for team management or for version control only. This is done by an integrator.

For details, see the Enabling Projects for Design Management chapter of *Allegro Design Management User Guide*.

Design Management – Project Management

Opens the Project Management dialog. When enabling, or after enabling a project for design management, you can make a user a Logical Integrator, Physical Integrator, or both using this dialog.

In this dialog, you can also define whether multiple users can check out the physical view of the design.

When you select the *Manage files under physical view as separate objects* checkbox, this enables file-level management and allows you to manage files under the physical view as separate objects.

For details, see the following:

- Defining Users as LDI or PDI section in *Allegro Design Management User Guide*
- File-Level Management sub-section in the Preparing Designs for Design Management section in *Allegro Design Management User Guide.*

Design Management – Show Dashboard

Use this option to display the dashboard in Allegro Design Management. The dashboard window shows the entire design and the state of all its subdesigns, their status, and team member ownership.

Design Management – Check Out

After the integrator has set up the shared area and assigned ownership rights for subdesigns in a project, you, as the designer, can access the project and start work on the subdesign you own.

Use this option to check out the required design objects such as subdesigns, or blocks when working with hierarchical designs or pages, or constraints when working with flat designs.

For details, see the Checking Out a Design, Subdesign, or Page section of *Allegro Design Management User Guide*.

Design Management – Check Out Hierarchy

Depending on your requirements and rights, you can check out just the design or the design along with all its subdesigns.

Check Out and Check Out Hierarchy can apply at any level in the design, whether it is root or a subdesign. Check Out hierarchy includes checking out all the child objects of the selected design.

For details, see *Allegro Design Management User Guide*.

Design Management – Download a Copy

An integrator in a team design environment can download a saved copy of a design object that was checked out and modified. The copy is downloaded from the SharePoint server.

Design Management – Check In

After the integrator has set up the shared area and assigned ownership rights for subdesigns in a project, you, as the designer, can access the project and start work on the subdesign you own.

Use this option to check out the required design objects such as subdesigns, or blocks when working with hierarchical designs or pages, or constraints when working with flat designs.

For details, see the Checking Out a Design, Subdesign, or Page section of *Allegro Design Management User Guide*.

Design Management – Save a Copy

In a team design environment, save a copy of the design object that you checked out and modified. The copy is saved to the SharePoint server.

Design Management – Undo Check Out

Undo the checkout of a design object in a team design environment. This returns the design hierarchy to its previous state and cleans up the work area.

For details, see the Rolling Back Changes or Undoing Check Out section in *Allegro Design Management User Guide*.

Design Management – Undo Check Out Hierarchy

Depending on your requirements and rights, you can check out just the design or the design along with all its subdesigns. After checking out the the hierarchy, you can undo the checkout.

For details, see *Allegro Design Management User Guide*.

Design Management – Update

Update merges the shared area changes with the local work area.

Design Management – Roll Back

Rolls back changes that are made to a design that was not checked out when working in a team design environment.

Designers may sometimes modify a drawing without checking out the design. In such cases, Allegro Design Management highlights this unauthorized modification.

In such cases, you can roll back the change.

For details, see the Rolling Back Changes or Undoing Check Out section in *Allegro Design Management User Guide*.

Design Management – Show Differences

When working with a cache-enabled project, this option displays cell-level changes between two versions of a schematic. For example, if you check out a schematic, add a new cell, then check the schematic in, using this option will display the differences between the two schematic versions.

Design Management – Version History

See the Allegro Pulse documentation for details.

Design Management – Labels

See the Allegro Pulse documentation for details.

Design Management – Delete

Deletes a managed page from the shared area. Managed pages are part of Allegro Design Management-enabled projects.

Design Management – Check Updates

Displays changes in the shared area before a notification from Allegro Design Management when working in a team design environment.

Design Management – Show Server Version

See the Allegro Pulse documentation for details.

Design Management – Advanced – Release Check Out

As an integrator, you can use the Release Check Out option to release the check-out done by any other user. You may need to do this, for example, if a team member is on leave or has

left the team. The LDI can only release the check-out of logical objects; the PDI can releasethe check-out of physical objects.

Design Management – Advanced – Check Out Saved Copy

In a team design environment, check out a copy of a design object that is modified and saved on the SharePoint server.

Design Management – Advanced – Switch to ECO Mode

Switches a design to ECO (ready for release) mode when working in a team design environment.

For cache-enabled project types such as board and highspeed, Allegro Design Management manages additional, internal global objects—shopping_cart and design_cache—using Library Revision Manager (LRM). Allegro Design Management operates in two types of modes: Work in Progress (WIP) and

The design_cache object is created when the integrator switches the design mode to ECO.

Design Management – Advanced – Refresh Policy File

Fetches all the changes from the site level when working in a team design environment.

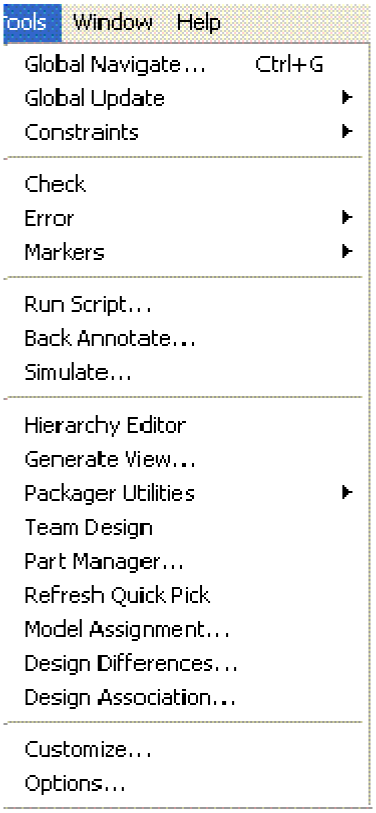
Important

> In case your project has object definitions that are not there in the site-level policy file, do not use the *Refresh Policy File* option. Doing so will overwrite the local policy file and objects defined based on the older policy file may no longer work.

Design Management – Advanced – Update Shopping Cart

Updates the shopping cart. The the shopping cart is only only in the data management mode.

Tools Menu



Tools – Global Find

Procedure

Displays the *Global Find* dialog box in which you specify a net or cell to be located in your design.

Tools – Global Update – Global Property Change

Opens the Property Change tabbed page of the Global Modification window. Use this page to change properties of components, pins, and nets across a design.

Tools – Global Update – Global Property Delete

Opens the Property Delete tabbed page of the Global Modification window. Use this page to delete properties of components, pins, and nets across a design.

Tools – Global Update – Global Component Change

Opens the Component Change tabbed page of the Global Modification window. Use this page to replace a component with a new component across a design.

Tools – Constraints – Edit

Displays the Constraint Manager, which you use to define, view, and validate electrical constraints in Design Entry HDL.

Constraint Manager is a cross-platform, workbook- and worksheet-based application used to manage high-speed electrical constraints across all tools in the Cadence PCB design flow.

Constraint Manager lets you define, view, and validate constraints at each step in the design flow, from design capture (in Design Entry HDL) to floorplanning (in Allegro SI) to design realization (in Allegro or Advanced Package Designer expert).

In Constraint Manager, you work with objects and electrical constraint sets (Electrical CSets). You define one or more Electrical CSets to capture your design requirements in the form of electrical constraints. You then assign the appropriate Electrical CSet to objects in your design, swapping Electrical CSet assignments (or re-defining the currently assigned Electrical CSet) as your design requirements change.

An Electrical CSet can be referenced by any number of objects in your design. Objects and Electrical CSets can be generic to the entire design or they can reference a specified net in the design.

For more information on using Constraint Manager, see the Allegro Constraint Manager User Guide.

Tools – Check

Command

Checks for connectivity problems and general errors in the current drawing and creates an error log and a markers file. After you check your design

- ❑ Choose *Tools – Error* (Tools – Error – Previous, Tools – Error – Next, Tools – Error – First, or Tools – Error – Last) to navigate error messages and locate violations on the schematic
- ❑ Choose Tools – Markers – Load to display the *Markers* control window, which gives you additional control over markers (such as viewing details about errors and filtering on errors).

You set electrical, graphic, name, and other checks in Tools – Options. In addition to these checks, *Tools – Check* also lists errors for the following:

- ❑ Objects partially outside the drawing boundaries
- ❑ Duplicate PATH properties
- ❑ Wires overlapping a component

Tools – Error – Next

Command

Displays the next error in the markers file, which is generated when you run Tools – Check. A blinking box appears on the schematic at the location of the next error in the markers file. A message appears in the Error Status Bar describing the error.

Tools – Error – Previous

Displays the previous error in the markers file, which is generated when you run Tools – Check. A blinking box appears on the schematic at the location of the previous error in the markers file. A message appears in the Error Status Bar describing the error.

Tools – Error – First

Displays the first error in the markers file, which is generated when you run Tools – Check. A blinking box appears on the schematic at the location of the first error in the markers file. A message appears in the Error Status Bar describing the error.

Tools – Error – Last

Displays the last error in the markers file, which is generated when you run Tools – Check. A blinking box appears on the schematic at the location of the last error in the markers file. A message appears in the Error Status Bar describing the error.

Tools – Error – Up

Displays the previous location of an error within a marker. A blinking box appears on the schematic at the previous location of an error within a marker. A message appears in the Error Status Bar describing the error.

Tools – Error – Down

Displays the previous location of an error within a marker. A blinking box appears on the schematic at the previous location of an error within a marker. A message appears in the Error Status Bar describing the error.

Tools – Markers – Load

Displays the *Markers* control window in which you can view messages about errors in your design and click an error message to see the corresponding elements highlighted in the design. The *Markers* control window also lets you

- View details on errors
- Filter to display only certain types of violations
- Delete markers as you fix them
- Load any number of markers files

The Markers control window is also displayed when you run Tools – Check and click *Yes* in the message box to view errors.

Note: You can use the keyboard shortcuts to delete markers (*Del* key) or undo changes (*Ctrl + Z*). You can also use the RMB menu to perform these actions – *Delete Marker* and *Undo Marker*.

Note: Before you can locate and correct violations on the schematic, you must first check your design using *Tools – Check*.

Tools – Markers – Packager

Displays the *Markers* control window in which you can view messages about the packaging errors in your design.

Tools – Markers – Netlisting

Displays the *Markers* control window in which you can view messages about the netlisting errors in your design.

Tools – Markers – Check

Displays the *Markers* control window in which you can view messages about the connectivity problems and general errors in your design detected by running the *Tools – Check* menu command.

Tools – Markers – Checkplus

Displays the *Markers* control window in which you can view messages about the errors reported by Design Entry Rules Checker in your design.

Tools – Markers – SheetImport

Displays the *Markers* control window in which you can view messages about the errors introduced in the design by importing a sheet or a block using the Import Design feature.

Tools – Markers – RF PCB Import

Displays the *Markers* control window in which you can view messages about the updates made to the design after the RF PCB Import process. The messages help you navigate to the changes or updates made to the schematic. You can load the marker file in DE-HDL and see all the changes listed in the *Markers* control window. On clicking a specific change, the relevant page is displayed and the relevant area is zoomed.

Load Markers File

Click to load all the marker files for the design. The marker files are displayed in the Markers File Name drop-down list in the Markers toolbar.

Markers File Name

Displays the name of the currently loaded marker file. Click on the drop-down to select another marker file.

All the marker files for the design are displayed in this drop-down list.

Tools – Script – Run Script

Procedure Command Examples

Lets you run Design Entry HDL commands in batch mode by specifying a script file. Scripts can call other scripts, and they can be interactive.

Examples

```
add ls04
$<Place the LS04
```

Add an LS04 to a drawing and use the mouse to place the component.

```
property
$<Choose the part to add a size to
size =
$<Type in the size you want
$<Place the property on the drawing
```

Add a size property to a part with a size specified at the time of entry.

```
rotate
$;Rotate the object until properly oriented
```

Rotate an object until the user enters a semicolon.

Tools – Back Annotate

Procedure Command

Displays a browser from which you choose a file containing the physical data with which you want to update the schematic.

Caution

Do not run backannotation if any other user who has write permissions is working on the design. Running backannotation when another user is working on the design results in incomplete backannotation.

Annotates designs with physical information produced by the design synchronization process, such as location designators, pin numbers, and physical net names on the design. Design Entry HDL edits the drawings you specify, adding the appropriate physical information, and then saves the drawing. Backannotation stops if any errors are detected during the process.

Note: In the backannotated design, properties added in Allegro are not visible on the Design Entry HDL canvas unless you specify a placeholder for each of them.

Physical net names also do not appear in the backannotated design unless there is a placeholder specified in the schematic. For example, When you have re-usable block that is used thrice in your schematic, three different physical nets are assigned in Allegro. This means that two physical net names are different from their logical net names. To view the physical net names in the backannotated design, place $PNN placeholders in Design Entry HDL during the design entry stage.

The annotated properties that Design Entry HDL adds are soft properties. This lets the Design Synchronization tool reassign the physical information each time the design is repackaged.

Use Tools – Options to set options that control property visibility and pin number placement

Tools – Simulate

Displays the *Start Verilog-XL* dialog box in which you perform the setup for Verilog-XL.

Tools – PIC [x]

Displays a vendor-specific dialog box for running the PIC Design Solution, a Design Entry HDL interface for designing PIC components using vendor libraries and tools. This menu command is available only when you choose to generate PIC output (use *Tools – Options –Output*).

Tools – Hierarchy Editor

Displays the Hierarchy Editor, which you use to create an expansion configuration before expanding a drawing and which provides a view of the drawing hierarchy. Using the Hierarchy Editor, you can

- ❑ Navigate the drawing hierarchy
- ❑ Trace signals throughout the design
- ❑ Find components
- ❑ Create design configurations with different simulation models
- ❑ Increase simulation capacity by partitioning your design across separate simulation processes

Tools – Generate View

Procedure Dialog Box

Displays the *Genview* dialog box, which you use to generate one design view from another.

Using *Tools – Generate View*, you can generate other views for the design from the existing ones. For example, you can automatically create a symbol for a schematic.

Tools – Packager Utilities – Bill of Materials

Displays the *Bill of Materials* dialog box for generating a Bill of Materials report. For additional information on generating a Bill of Materials, see *Design Synchronization and Packaging User Guide*.

Tools – Packager Utilities – Electrical Rules Check

Displays the *Electrical Rules Check* dialog box, which you use to set the electrical checks performed during packaging. For additional information on electrical rules checking, see *Design Synchronization and Packaging User Guide*.

Tools – Packager Utilities – Netlist Reports

Displays the *Netlist Reports* dialog box, which lets you generate and view netlist information for your design. For additional information on generating netlist reports, see *Design Synchronization and Packaging User Guide*.

Tools – Refresh Quick Pick

Refreshes the Quick Pick Browser

Tools – Design Differences

Procedure

Displays the *Design Differences* dialog box, which lets you compare the design differences between the schematic and the physical board layout. Before you compare the schematic and physical board design, you can also update the physical board view or the packaged schematic view with any design changes.

You must run Design Differences before running Design Association to apply connectivity changes.

For more information on design differences,

➤ Click *Help* in the Design Differences dialog box,

Tools – Design Association

Procedure

Displays the Design Association window, which lets you apply connectivity changes on the physical board to the schematic. For more information on Design Association, click *Help – Documentation* in the Design Association window.

Tools – Options

Command

Establishes Design Entry HDL editor default settings. Use the Project Manager setup to specify basic information for your entire design (design name, library names, and locations).

Tools – Part Manager

Command

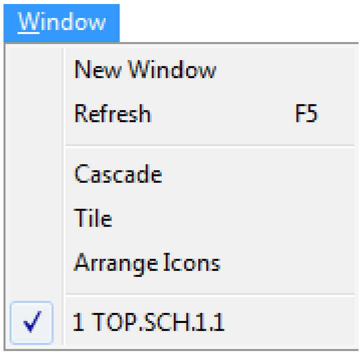
Launches the Part Manager window. Use this window to view status of part instances and update part instances on a schematic with an appropriate row in the associated part table file (ptf).

Tools – Model Assignment

Command

Launches the Part Manager window. Use this window to view status of part instances and update part instances on a schematic with an appropriate row in the associated part table file (ptf).

Window Menu



Window – New Window

Command

Opens another window for the active design so you can look at different perspectives of the same drawing or open different drawings simultaneously. Any operations you perform in either window appear in both copies of the drawing.

Note: With more than one window, you can:

- zoom windows independently to focus on different sections of a design
- draw or route wires between windows
- use one window as a global view of the original design

When switching between windows, activate a window by clicking in the window.

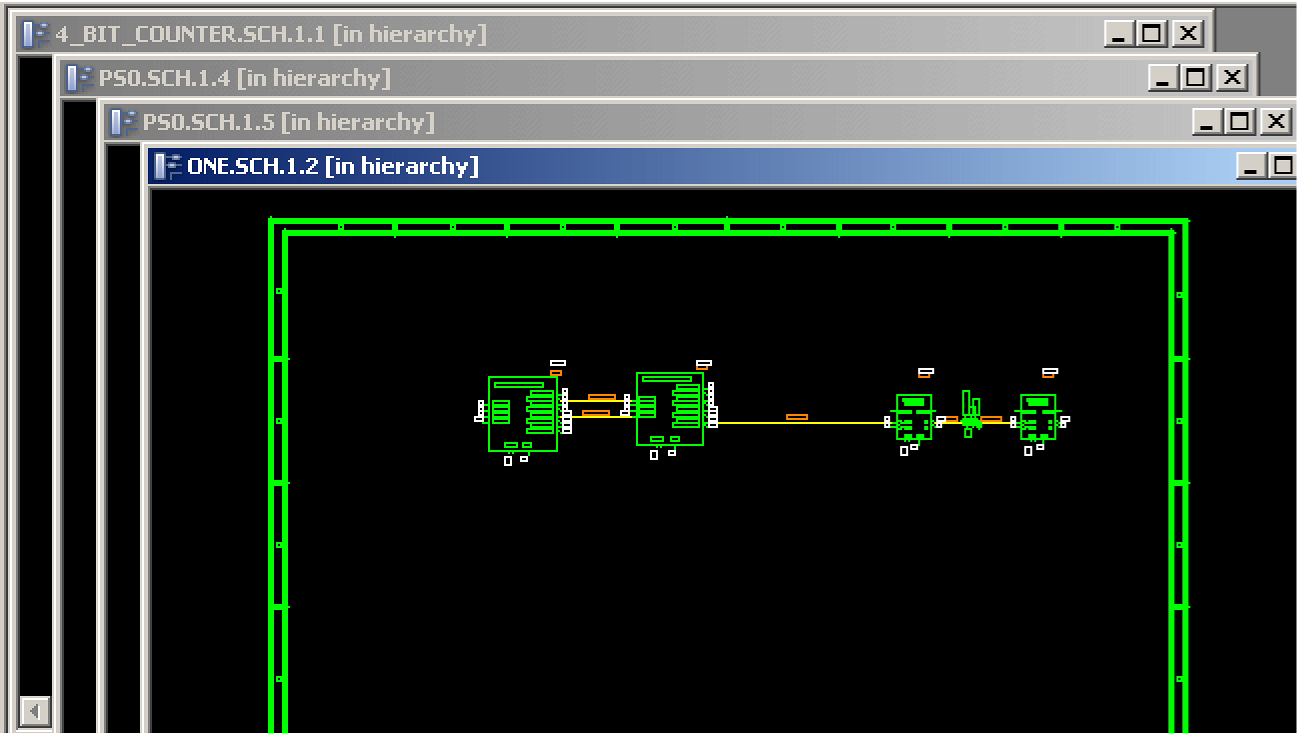
Window – Refresh

Command

Refreshes the active window.

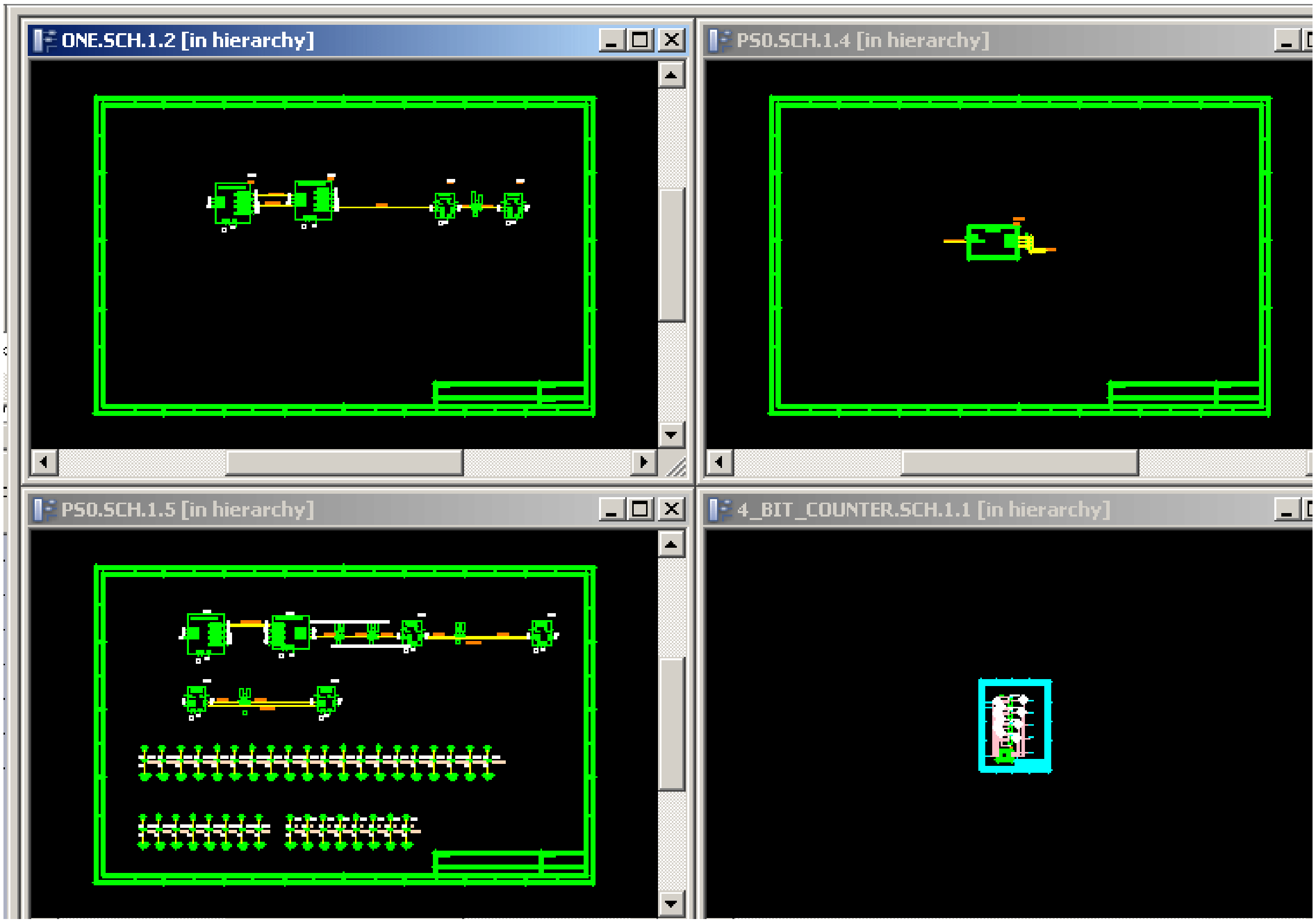
Window – Cascade

Arranges windows so they overlap with only title bars displayed.


4_BIT_COUNTER.SCH.1.1 [in hierarchy]
PS0.SCH.1.4 [in hierarchy]
PS0.SCH.1.5 [in hierarchy]
ONE.SCH.1.2 [in hierarchy]

Window – Tile

Arranges windows as non-overlapping tiles.


ONE.SCH.1.2 [in hierarchy]
PS0.SCH.1.4 [in hierarchy]
PS0.SCH.1.5 [in hierarchy]
4_BIT_COUNTER.SCH.1.1 [in hierarchy]

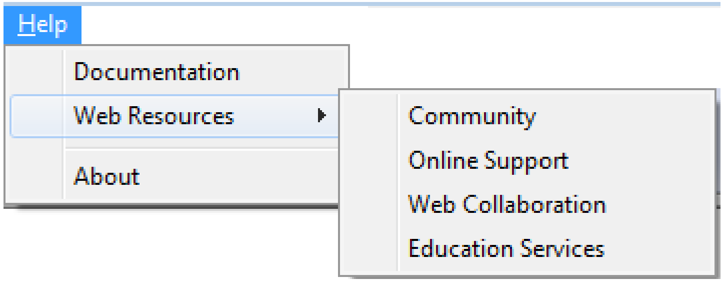
Window – Arrange Icons

Arranges icons at the bottom of the Design Entry HDL window.

Window – *drawing_name*

For each window, displays a list of drawings that were edited in that window.

Help Menu



Help – Documentation

Opens the Help System page for Allegro Design Entry HDL. From this page, you can access the help documents and utilities available for Design Entry HDL and related tools. This web page contains four tabbed pages:

- **Documentation:** Displays links to the available documentation for Design Entry HDL and related tools.
- **Release Info:** Displays the following documents for Design Entry HDL:
  - What's New in Design Entry HDL
  - Known Problems & Solutions
  - Allegro Platform Migration Guide
  - Allegro Platform System Requirements
  - Frequently Asked Questions .
- **Tutorials:** Displays the list of tutorials that help you in getting started quickly with Design Entry HDL and related tools.
- **Demos:** Lists the various multimedia demos that you can run to learn the features in Design Entry and SPB flows.

Help – Web Resources – Community

Opens the Cadence community web site (http://community.cadence.com) in a web browser. This web site provides information on all Cadence products.

Help – Web Resources – Online Support

Opens the Cadence Online Support web site in a web browser. Cadence Online Support is the online customer support web site for Cadence software users.

Help – Web Resources – Web Collaboration

Launches the SpaceCruiser server site. SpaceCruiser is a secure client/server software solution, based on industry standards, for desktop sharing and Web conferencing.

Help – Web Resources – Education Services

Opens the Cadence Education Services web site in a web browser. The Education Services web site provides information on training courses and related services from Cadence.

Help – About

Displays the About dialog box for Allegro Design Entry HDL. It shows the version number of the installed product, and the information about copyright and patents.

Change Mode

Toggles the pin mode before you place the pin on the block.

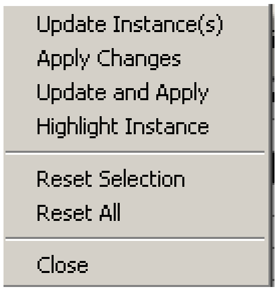
You can also toggle the pin mode by pressing *Ctrl* and clicking the left mouse button in a two-button mouse or clicking the middle mouse button in a three-button mouse.

Note: You cannot toggle the pin mode after you place the pin on the block.

Select Object

Brings you out of command mode and lets you select objects.

Part Manager Menu



Options – Update Instance(s)

Procedure

Updates an undefined part instance in the design. You can choose an appropriate row with which you want to update the selected part instance. The status of the selected row in the grid changes from “NON MATCHED” (red) to “MATCHED” (green) with a shaded background. The shaded background signifies that the part instance has been modified in the memory. However, the change is yet to be reflected on the schematic

Alternatively, you can right-click a row with a “NOT MATCHED” (red) part status on the Part Manager grid, and select *Update Instance(s)* from the pop-up menu.

Options – Apply Changes

Procedure

Applies changes to the schematic. After using the *Update instance(s)* command, you need to apply the changes for the changes to take effect on the schematic.

Alternatively, you can select the updated row and click the *Apply* button

Options – Update and Apply

Updates a "NON MATCHED" part instance in a design with a valid ptf row, and applies changes to the schematic.

Alternatively, you can right-click a row with a "NON MATCHED" part status on the Part Manager grid, and select *Update and Apply* from the pop-up menu.

Options – Highlight Instance

Procedure

Highlights a specific part instance on a schematic from within the Part Manager grid. This command is available for single rows only.

Alternatively, you can right-click a row in the Part Manager grid, and select *Highlight Instance* from the pop-up menu.

Group – Assign Signal Model [A]

Displays the *SI Model Assignment* dialog box that lets you assign signal models to a group of devices.

Options – Reset All

Procedure

Helps you revert back to the original schematic values of all the Part Manger rows. The *Reset All* command helps you undo changes made in multiple rows of the Part Manager grid. It is particularly useful when you update a row incorrectly and want to undo the update operation.

Options – Reset Selection

Procedure

Helps you revert back to the original schematic values of the selected Part Manager row. The *Reset All* command helps you undo changes made to a single row in the Part Manager grid.

Add Split

Procedure

The Add Split option enables you to attach a property (RFSPLIT) to the wires selected. If a wire is attached with this property, then the logic group is broken where the property is added (one big logic group is split into two logic groups).

Disband

Procedure

The Disband option enables you to remove the RFGROUP property from each RF component for the specific group.

Exclude

Procedure

Exclude enables you to remove the property for selected objects (RFGROUP for RF components or RFSPLIT for wires).

View – Show Hierarchical Path

Procedure

Displays the complete hierarchical path of all the part instances in the Part Manager grid. This command adds a new column, titled Hierarchical Path, at the end of the Part Manager grid. This column displays the canonical path of each part instance.

Alternatively, you can right-click anywhere in the Part Manager window outside the grid and the Design Part Names list, and select *Show Hierarchical Path* from the pop-up menu.

The menu command changes to *Hide Hierarchical Path* when the hierarchical path column is added to the Part Manager grid. You can hide the column by selecting this command.

View – Select All

Procedure

Selects all the rows from the Part Manager grid for a given part. This command is normally used when you want to update multiple part instances simultaneously.

Alternatively, you can right-click any row in the Part Manager grid, and choose *Select All* from the pop-up menu.

Tools – Customize

Dialog Box

Displays the *Customize* dialog box that you can use to customize the User Interface. You can customize the following UI elements – Toolbars, Buttons, Commands, Menus, and Keys.

Toolbars	Use this tab to add new toolbars and select which toolbars to display.
Buttons	Use this tab to add or remove buttons from toolbars, and moving buttons from one toolbar to another.
Commands	Use this tab to add new commands and associate keyboard keys and menu items to them.
Menus	Use this tab to add new menus and menu items.
Keys	Use this tab to associate keyboard key combinations to Design Entry HDL commands or your own commands.

Open

Opens a module in the Design Entry HDL window for editing. You can also click or double-click the module in the Hierarchy Viewer window to open it in the Design Entry HDL window.

Open in New Window

Opens a module in a new Design Entry HDL window.

Select Instance

Highlights the instance of the selected module on the parent schematic with a blinking box. The *Select Instance* option is disabled for the top-level module.

Go To Page

Allows you to jump to a page/symbol in a design.

Hide Sheet Numbers

Shows or hides sheet numbers from appearing in the Hierarchy Viewer window.

Hide Instance Names

Shows or hides instance names from appearing in the Hierarchy Viewer window.

Refresh Hierarchy

Updates the tree structure in the Hierarchy Viewer window with any changes made to the design, such as deleting or adding a new module to the design.

Module Order – Exclude Occurrence

Excludes only the current occurrence of the module. If a cell has been excluded using module ordering or `xmodules.dat`, the sheet number for the cell is not shown.

Module Order – Exclude All

Excludes all occurrences of the module.

Module Order – Include Occurrence

Includes only the current occurrence of the module.

Module Order – Include All

Includes all occurrences of the module.

Module Order – Hide Excluded Modules

Hides excluded modules.

Module Order – Excluded Modules

Displays a list of all excluded modules.

Module Order – Reset Module Order

Clears all exclusions and inclusions.

Allow Docking

Disables or enables docking of the Hierarchy Viewer window.

Hide

Shows or hides the Hierarchy Viewer window.

Refresh Hierarchy

Updates the tree structure in the Hierarchy Viewer window with any changes made to the design, such as deleting or adding a new module to the design.

8

Dialog Box Help

This section describes the various dialog boxes of the Allegro Design Entry HDL schematic editor.

Add Component-Library View

Command

Use this dialog box to place designs and individual components on your drawing. The **Library View** page lets you display components by library. Design Entry HDL allows you to select multiple components in the Add Component dialog box (Part Information Manager) and view a combined list of physical components in the Physical Part Filter. Design Entry HDL allows multiple selection of components only when the Physical Part Filter is displayed.

Library Identifies the library whose components you want to list in the **Cells** box.

Filter Lets you narrow the list of components or designs using wildcard characters:

* matches any text string.
? matches any single character.

Cells Lists components and designs.

New Window Displays a new Part Information Manager dialog box. This is useful if you want to view the contents of more than one library at once.

Physical Displays the Physical Part Filter to add a component with physical information. To open the Physical Part Filter every time you choose *Component – Add*, choose *Tools – Options* and click on the General Tab. In the General page, select the Show PPT Browser check-box.

Close Closes the Part Information Manager dialog box.

Note: The **Library View** page is displayed by default when you open this dialog box. To display the **Category View** page by default, select the **Show Category View (Add)** check box in the General page of the **Design Entry HDL Options** dialog box (choose **Tools – Options**).

Add Component-Category View

Command

Use this dialog box to place designs and individual components on your drawing. The **Category View** page lets you display categories of components arranged hierarchically.

You can select a category and view all the physical components in the selected category. This allows you to select the exact physical component and place it in the logical design. To do this, select a category and click Right Mouse Button to select **Select All Sub Parts** and click on **Physical**. Design Entry HDL allows selection of all parts in a category only when the Physical Part Filter is displayed.

New Window Displays a new **Part Information Manager** dialog box. This is useful if you want to view the contents of more than one library at once.

Physical Displays the Physical Part Filter to add a component with physical information. To open the Physical Part Filter every time you choose *Component – Add*, choose *Tools – Options* and click on the *General* Tab. In the General page, select the *Show PPT Browser* check-box.

Close Closes the Part Information Manager dialog box.

Note: The **Library View** page is displayed by default when you open this dialog box. To display the **Category View** page by default, select the **Show Category View (Add)** check box in the General page of the Design Entry HDL Options dialog box (choose **Tools – Options**).

Physical Part Filter

Use this dialog box to include or change physical information of a component from a Physical Part Table (PPT) on the schematic. The Physical Part Filter lets you select a physical component based on the logical component you select in Part Information Manager.

Design Entry HDL allows you to select multiple components in the **Add Component** dialog box (Part Information Manager) and view a combined list of physical components in the Physical Part Filter. To select multiple components in the Library view, press SHIFT+Click to select components in the list. To select components randomly from the list, press CTRL+click. Design Entry HDL allows multiple selection of components only when the Physical Part Filter is displayed.

You can also select a category and view all the physical components in the selected category. This allows you to select the exact physical component of your choice and place it in the logical design. To do this, select a category and click the Right Mouse Button to select **Select All Sub Parts** and click on **Physical**. Design Entry HDL allows selection of all parts in a category only when the Physical Part Filter is displayed.

Part Names

Displays the part names. You can select multiple parts in the Part Information Manager dialog box (both with Library and category view).

Column headings

Lists physical property names from left to right in the order you specify in the Property Options dialog box.

The **Filter** field on top of each column heading allows you to filter physical property values based on the string you enter.

Example: If you have ten rows appearing in the Physical Part Filter and you want to filter out and use only those physical parts with part numbers starting with 1, enter 1* in the *Filter* field and press enter. Design Entry HDL displays only those physical part table rows with part number starting with 1.

Use Case Sensitive Filtering

Select this check box to use case sensitive filtering of physical properties from a PPT.

For Example: Consider that the PPT for the selected part has two rows and the values for the JEDEC_TYPE property are C123 and c123. If you select Case Sensitive Filtering, type C* in the filter, and press enter, The Physical Part Filter will display only that row which has C123 as the value for the JEDEC_TYPE property.

Do Not Annotate Property With Optional Value

Select this check box if you do not want a key property having a value same as the optional value specified in the `.ppt` file to be annotated.

For example, say the optional value of a key property is '–' and you do not want to annotate it, select this check box.

Show All Primitives

Select this check box to load all the primitives from the `chips.prt` file.

If you do not select this, only those primitives for which BODY_NAME and PART_NAME match are loaded. In case there are no primitives for which the two match, all the primitives are loaded.

Reload PPT

Click this to reload the PPT file.

This is useful when you have edited the PPT file and want to reload the PPT file in the Physical Part Filter.

Reset Filters

The filters are set to the preset values in the `pptoptions.dat` file(s) being used currently.

In the absence of the `pptoptions.dat` file(s), the filters are set to *.

Options…

Click this to display the Property Options dialog box. The Property Options dialog box allows you to define settings for annotating physical properties on the schematic.

Close

Closes the Physical Part Filter.

View Open

Use this dialog box to open a component or design for edit.

Library Identifies the library whose components you want to list in the scroll box.

Cell If you know the name of a component or design, you can type it in this box.

View Specifies **Schematic** (logical), **Symbol** (symbolic representation), or the view name based on the view you want to open.

Version Displays the version number of a schematic or symbol representation (the default is **1**).

Page Displays the page number of a schematic (the default is **1**).

Search Stack Displays the **Search Stack** dialog box, which you use to define the search order of libraries and designs.

Open Opens the specified drawing. This button is enabled when you enter or select a valid drawing name.

Cancel Closes the **View Open** dialog box.

New Window Opens the specified drawing in a new viewport (by default it is selected).

Browse Opens the specified drawing but does not close the **View Open** dialog box.

Use this option to open multiple drawings while the dialog box is open.

Filters >> Lets you narrow the list of components or designs.

Cell Name-Use wildcard characters:

***** matches any text string.
? matches any single character.

View Name-Limits the views listed by matching a pattern against the view name.

View Type-Limits the views listed to **Schematic** for logic views, **Symbol** for symbol views, or **All** for a complete list of components.

You can also click on a component, press the right mouse button and choose **Open**. The views for the component are displayed. Click the view that you want to open. For example, click on a symbol view (sym_n) to open the symbol for the component.

View Save As

Procedures Command

Saves a component or design under a new name in the library that you specify.

Library Identifies the library whose components you want to list in the scroll box.

Cell If you know the name of a component or design, you can type it in this box.

View Specifies **Schematic** (logical), **Symbol** (symbolic representation), or the view name based on the view you want to open.

Version Displays the version number of a schematic or symbol representation (the default is **1**).

Page Displays the page number of a schematic (the default is **1**).

Search Stack Displays the **Search Stack** dialog box, which you use to define the search order of libraries and designs.

Save Saves the specified drawing.

Cancel Closes the **View Save As** dialog box.

Filters >> Lets you narrow the list of components or designs.

Cell Name-Use wildcard characters:

***** matches any text string.
? matches any single character.

View Name-Limits the views listed by matching a pattern against the view name.

View Type-Limits the views listed to **Schematic** for logic views, **Symbol** for symbol views, or **All** for a complete list of components.

View Remove

Command

Use this dialog box to delete a component or design from the library that you specify.

Library Identifies the library whose components you want to list in the scroll box.

Cell If you know the name of a component or design, you can type it in this box.

View Specifies **Schematic** (logical), **Symbol** (symbolic representation), or the view name based on the view you want to open.

Version Displays the version number of a schematic or symbol representation (the default is **1**).

Page Displays the page number of a schematic (the default is **1**).

Search Stack Displays the **Search Stack** dialog box, which you use to define the search order of libraries and designs.

Remove Deletes the specified drawing.

Cancel Closes the **View Remove** dialog box.

Filters >> Lets you narrow the list of components or designs.

Cell Name-Use wildcard characters:

***** matches any text string.
? matches any single character.

View Name-Limits the views listed by matching a pattern against the view name.

View Type-Limits the views listed to **Schematic** for logic views, **Symbol** for symbol views, or **All** for a complete list of components.

Search Stack

Procedures Command

Use this dialog box to view or edit the search stack ordering.

Search Stack Lists available libraries in the order Design Entry HDL searches them for drawings to edit or components to add to your drawing.

Browse Displays Part Information Manager. You must first click on a library listed under *Search Stack*.

Edit>> Expands the Search Stack dialog box so you can add and remove libraries in the search stack or change their order.

If you add or remove libraries this way, the changes are not saved to your project when you exit. To make permanent changes to your project, use Project Manager Setup.

<Add Adds a specified library to the search stack.

Position Specifies whether to add a library to the top or bottom of the search stack.

< Add All Adds all the available libraries to the search stack.

Up Moves the selected library one position up in the search stack.

Down Moves the selected library one position down in the search stack.

Ignore> Removes the selected library from the search stack.

Ignore All> Removes all but one libraries from the search stack. The top-most library continues to be in the search stack

Add Library Opens the Select Folder dialog where you can choose a library to be added to the search stack. When added, the new library shows in the search stack.

Close Closes the **Search Stack** dialog box.

Location Shows the path to the library selected in the list.

Attributes

Procedures   Command

Use this dialog box to view, add, delete, and modify most schematic properties on components, pins, or wires (nets).

This dialog box contains different types of information depending on the selected object and the editing mode:

Procedures

- Adding Properties
- Displaying and Modifying Property Attributes
- Making an Attributes File

Cadence Product Choices

You can choose a product suite in which you want to run Design Entry HDL. Changing product suites allows you to access components that are not available in the current product suite. The product suites available for use are displayed in the list.

How to Access

The Cadence Product Choices dialog box is invoked when:

- you are using the tool for the first time; and on all subsequent invocations unless you specify the default choice.
- you choose *File – Change Product*.

Setting a Default Product Choice

To prevent the Cadence Product Choices dialog box from appearing every time you run Design Entry HDL, complete the following steps.

1. Select the product suite to be used as the default choice.
2. Select the *Use as Default* check box to invoke the selected product suite every time you invoke Design Entry HDL.

   Selecting the *Use as Default* check box writes the product choice in registry. The Design Entry HDL interface changes to reflect the selected product suite and will open with this product suite until you change the default setting.
3. Click *OK*.

To change the default product suite:

1. Choose *File – Change Suite* in Design Entry HDL.
2. Select the required product suite from the list of choices in the Cadence Product Choices dialog box.
3. Select the *Use as Default* check box
4. Click *OK*.

Specifying Product Choice from Command Line

If you invoke Design Entry HDL from command line, you can use the `-product` option to prevent the Cadence Product Choices dialog box from appearing every time.

Syntax

```
concepthdl -product "license_string"
```

You can choose one of the following license strings:

- `concept_hdl_expert`
- `concept_hdl_studio`
- `allegro_performance`
- `Allegro_Design_Editor_620`
- `pcb_librarian_expert`
- `Allegro_Frontend_PCB_Solution`
- `Allegro_Venture_SDA`
- `Allegro_Enterprise_SDA`
- `Allegro_Enterprise_PCB_Designe`
- `Allegro_Venture_PCB_Designer`

Note: License strings (suite names) are not case-sensitive.

Example

```
concepthdl -product concept_hdl_studio
```

In addition to the product name, you can also specify product option/feature in the command as illustrated in the following example:

```
concepthdl -product concept_hdl_studio Allegro_TeamDesign_Auth_Option
```

Following are the product options available:

- `Allegro_TeamDesign_Auth_Option`

Disabling License Check

To ensure that only the product suites for which you have licenses available, are displayed in the Cadence Product Choices dialog box, the application checks with the license server for available licenses. The process of populating the dialog box with the list of available licenses takes some time.

However, if the time taken for displaying the Cadence Product Choices dialog box is high, you can use the CDS_IGNORE_LIC_FEATURE environment variable, with its value set to TRUE, to disable the procedure of checking for the available licenses. Using this variable ensures that the dialog box appears instantly, but displays all the licenses using which you can launch Design Entry HDL. From the list, you need to select the product suite for which you have the license available. For information on the available licenses, contact your license administrator.

Text Input

Procedures Commands Related Info

Use this dialog box to enter signal names, note text, and block pin names.

Signal Names | Notes | Block Pin Add | Block Pin Names	The title of this dialog box and the label for the entry area change depending on whether you choose: *Wire – Signal Name* *Text – Note* *Block – Add Pin* *Block – Rename Pin* Enter signal names, note text, a URL, and block pin names in the entry area, one per line.
Close	Closes the dialog box and cancels any input made to the entry area.

Queue	Attaches the top line of text to the cursor for you to place on the drawing. After being placed, the top line is removed from the queue, and the next line is attached to the cursor.
Select	Attaches the text you select to the cursor for you to place on the drawing. While the text is selected, you can place it in multiple locations on the drawing.

Procedures

- Naming a Signal
- Adding Block Pins
- Renaming Block Pins
- Adding Text

Commands

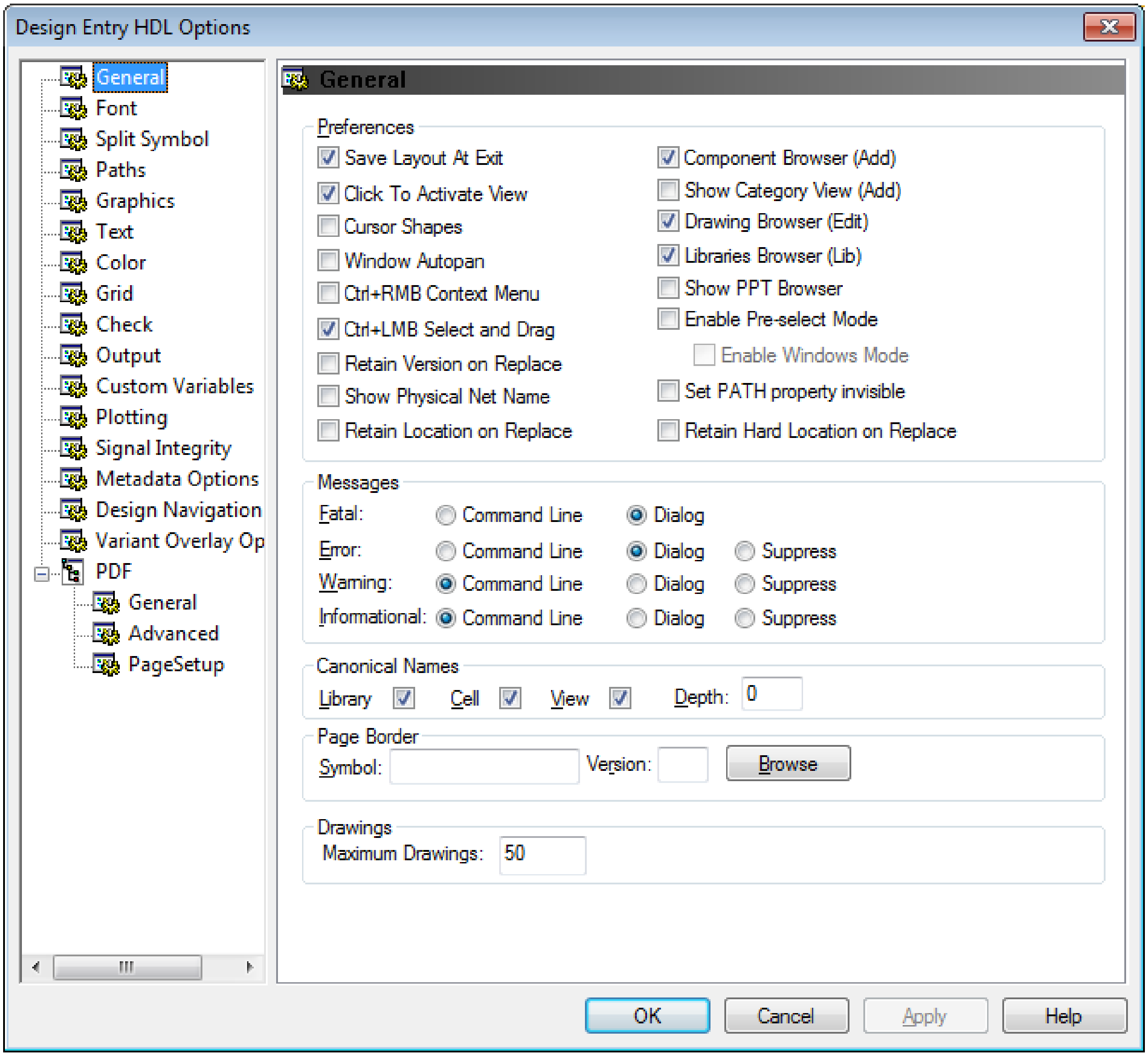
- Signame
- Bpadd
- Bprename
- Note

Related Info

Change Mode

Design Entry HDL Options

You can invoke the *Design Entry HDL Options* dialog box by selecting the *Tools – Options* menu option. This dialog box contains tabs that you can use to set various options for the Design Entry HDL schematic editor.



Preferences

☑ Save Layout At Exit
☑ Click To Activate View
□ Cursor Shapes
□ Window Autopan
□ Ctrl+RMB Context Menu
☑ Ctrl+LMB Select and Drag
□ Retain Version on Replace
□ Show Physical Net Name
□ Retain Location on Replace
☑ Component Browser (Add)
□ Show Category View (Add)
☑ Drawing Browser (Edit)
☑ Libraries Browser (Lib)
□ Show PPT Browser
□ Enable Pre-select Mode
□ Enable Windows Mode
□ Set PATH property invisible
□ Retain Hard Location on Replace

Messages

Fatal:	○ Command Line	● Dialog	
Error:	○ Command Line	● Dialog	○ Suppress
Warning:	● Command Line	○ Dialog	○ Suppress
Informational:	● Command Line	○ Dialog	○ Suppress

Canonical Names

Library ☑ Cell ☑ View ☑ Depth: 0

Page Border

Symbol: Version: Browse

Drawings

Maximum Drawings: 50

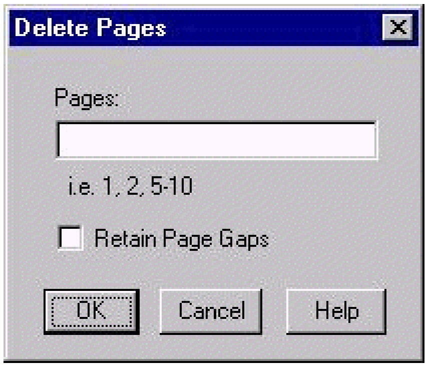
General Use to set options for paths, preferences, and message display.

Font	Use to specify font attributes for different types of schematic text objects.
Split Symbol	Use to specify settings for creating split symbols manually or using the auto-distribution method.
Paths	Use to set paths for various files read by Design Entry HDL.
Graphics	Use to set options for wires, wire connections, and taps.
Text	Use to set options for text and properties.
Color	Use to set colors for objects and background.
Grid	Use to set options for grids.
Check	Use to set options for error checking.
Output	Use to set options for saving drawings.
Custom Variables	Use to define custom variables for the current project.
Plotting	Use to set options for Windows plotting.
Signal Integrity	Use to set default pin models.
Metadata Options	Use to set options for generating schematic metadata and launching Component Revision Manager.
Design Navigation	Use to display page names along with the block names and page number ranges under each block in the design.
Variant Overlay Options	Use to set options to display selective information in a variant: display or hide the DNI property, cross out the DNI components, or show or hide all properties of DNI components on the canvas.
PDF	Use to set the preferences to specify what information is to be exported to the PDF document of the design. You can also control the default visible information at the time of document loading by setting the publishing preferences. For details, see *Allegro Design Publisher User Guide*.

Delete Pages

Procedure Command

Use this dialog box to delete a page or a set of pages from a schematic.



Field	**Description**
Pages	Indicates the pages you want to delete from the schematic. You can specify an explicit number or a range of numbers to be deleted. For example, 1,2,3,5-7 is a valid range. Spaces are not allowed between page numbers.
Retain Page Gaps	Retains the physical page numbers of the pages affected by the deletion. This option is deselected by default. Selecting this option will create page gaps for the pages you delete.

For more information, refer to the Performing Page Management Operations section of the Allegro Design Entry HDL User Guide.

Design Entry HDL Options - Graphics

Command   Related Info

Use these setup options to establish Design Entry HDL editor default settings for wires, wire connections, and taps.

Wires	
Add and **Move**	Draws wires that you add and move as **Orthogonal** or **Direct**

Auto Route On Move
Automatically routes wires around objects when you move a component in the drawing.

Note: This option is applicable only when Move is set in the Direct Mode and not in the Orthogonal mode.

Auto Heavy If Bus Name
Automatically thickens a wire when you attach a bus signal name to it.

Auto Name on Tap
Automatically inserts the specified tap symbol, bits, and wire names of bus taps when you use the tap command. Provides a shortcut for creating a tap (Wire – Bus Tap) specifying tap values (Wire – Bus Tap Values) and naming a tap (Wire – Bus Name).

You must specify tap bits when using the tap command (see Wire – Bus Tap Values).

Tap Symbol
Specifies the tap symbol to be used in a schematic.

Dots

Open or **Filled**
Adds open or filled dots at wire connections.

Auto Dot at Intersection
Automatically displays dots at wire connections.

Logic Dot Radius
Adjusts the diameter of dots at wire connections in schematic drawings and published PDF.

The valid values range from 1 to 40. For any value greater than 40, DE-HDL retains the last valid value set by the user. In case of PDF Publisher, any value from 1 to 5 is rendered to 6 in the published PDF.

Symbol Dot Radius
Adjusts the diameter of dots at wire connections in symbol drawings.

The valid values range from 1 to 40. For any value greater than 40, DE-HDL retains the last valid value set by user.

Note: Design Entry HDL displays the change in the logic dot radius and symbol dot radius only on plots and plot previews.

Design Entry HDL Options - Text

Command Related Info

Use these setup options to establish Design Entry HDL editor default settings for text and properties.

Text

Size

Specify the size of text (property name, property value, signal name or note) in the plotted schematic. The default value is 0.072 inches.

The minimum text size is 0.008 and the maximum text size is 1.746. The text size will always be a multiple of 0.00174 inches.

If you specify the logic grid as 0.1 inches and the text size as 0.1 inches, and enter text (property name, property value, signal name or note), Design Entry HDL places the text exactly between the two grid rows.

If you modify the text size, the new text size will apply only to the text that you add after the change. The size of the text that is already existing in the schematic will not change if you modify the text size.

For example, suppose that a signal name RESET has the text size of 0.072. If you modify the text size to 0.96, the text size of signal name RESET will not change. If you add another wire in the schematic and name it as DATA, the text size of the signal name DATA will be 0.96 (the new text size).

To change the size of the text in a schematic page, use the textsize console window command. For example, to change the size of all the text in a schematic page to 0.96 inches, create a group, say A, that covers all the objects in the schematic page and enter the following command in the console window:
textsize 0.96 A.

Justification

Justifies text **Left**, **Center**, or **Right**.

Upper-case Input

Displays text as all caps.

Text Change Editor

Specifies the text editor that Design Entry HDL displays for certain functions.

Properties

Visibility

Controls the way properties are displayed–**Invisible**, **Name** only, **Value** only, or **Both** name and value.

Pin Property Visibility

Controls the visibility of symbol pin properties when the symbol/component is instantiated on the schematic. **Invisible** does not display the symbol pin properties. **Defined by Component** makes pin properties visible or not depending on how property visibility is defined on the symbol.

Example: Let us say that the symbol PQR.SYM.1.1 has the property NO_SWAP_PIN = TRUE attached to a pin on it. If the visibility for this property is set to **Name** in the symbol pin, and you select **Defined by Component** in Design Entry HDL, NO_SWAP_PIN will be visible on component PQR after it is instantiated on a schematic in Design Entry HDL. If you select **Invisible**, NO_SWAP_PIN will not be visible on the schematic, but will be visible in the symbol.

Pin Number Size

Adjusts the size of the pin number displayed on the schematic to be larger or smaller. The unit is in inches. The pin number size is not related to Text Size you specify in this dialog box.

Rotate Vertical Pin Numbers During Backannotation

Automatically rotates pin numbers that are attached to vertical pins.

Design Entry HDL does not do this if the design has already been backannotated. Previous pin text orientations will not change or get updated. The part must be sectioned to remove pin number text, repackaged, and then backannotated again.

Auto Path Properties On Components

Automatically attaches a PATH property to an added part.

Retain Deleted Symbol Properties As Instance Properties	Retains a property deleted from the symbol drawing as an instance property.
Power Property Visibility	Controls the visibility of power pins properties created through the Assign Power Pins dialog box on the schematic–**Invisible**, **Name** only, **Value** only, or **Both** name and value. **Value** is the default selection.

Design Entry HDL Options-color

Command   Related Info

Use these setup options to establish Design Entry HDL editor default color settings for objects and background.

New Drawing Objects	Sets the graphic and background colors, as specified in Graphic Color and Background Color boxes, for the new drawing objects.
Image Capture	Sets the graphic and background colors, as specified in Graphic Color and Background Color boxes, for an image of the schematic captured and placed on the clipboard.
Graphics Color	Specifies default colors for the listed objects (left) and lists available colors from which you can choose to change the default (right).
Background Color	Lists colors from which you can choose to change the default for the drawing area.

Design Entry HDL Options-Grid

Command   Related Info

Use these setup options to establish Design Entry HDL editor default settings for grids.

Type

Defines the grid type:

Decimal–Bases drawings on the decimal system (500 units per physical inch).

Fractional–Bases drawings on 400 units per inch. Components will appear 25 percent larger.

Metric– Bases drawings on the metric system (20 units per millimeter; 508 units per inch).

You should use the same grid type for your schematics as used while creating symbols for components instantiated in the schematics. If you use different grid types, the symbols can get off-grid and cause connectivity problems.

Logic Grid

Defines the grid for schematic drawings.

Symbol Grid

Defines the grid for symbol drawings

Document Grid

The documentation grid is currently not supported in DE-HDL.

Show

Displays or hides the grid.

Style

Displays the grid as **Dots** or dashed **Lines**.

Size

Adjusts the grid size to be smaller or larger.

Multiple

Displays every nth grid line to define where objects can be placed so that pins do not fall off-grid. This ensures the correct connectivity of wires and symbols.

Design Entry HDL Options-Check

Command   Related Info

Use these setup options to establish Design Entry HDL editor default settings for error checking.

Check on Write Runs a check whenever you save the design. Errors are recorded in `cp.mkr` and `netlister.mkr`.

Electrical Checks

Missing Pins Checks for pin properties that are no longer attached to pins.

Unconnected Wires Checks for unnamed wires connected to only one pin (NC wires) and for named nets not connected to any pins.

Shorted Pins Checks for pins attached to more than two wires.

Power-Local Signal Short Checks for local signals connected to power symbols whose names are different from the value of the `HDL_POWER` property of the power symbol.

Voltage on Power Symbols Checks for the presence of the VOLTAGE property on an HDL_POWER symbol. If the VOLTAGE property is not present, a warning message is displayed.

Note: If you change the VOLTAGE property on an HDL_POWER symbol in Constraint Manager, the changed value will not appear on the HDL_POWER symbol in the Occurrence Edit mode until you backannotate it to the schematic canvas.

Graphics Checks

Symbols at Same Location Checks for overlaid components.

Hidden Wires Checks for wire segments hidden by portions of components.

Pins Near Wire Ends

Checks for wires that do not quite contact pins. Design Entry HDL generates an error message if the distance between a wire end and a pin falls below a minimum distance called the threshold.

Notice that the *Default* option button is selected. In this option, the threshold is calculated based on an internal algorithm. This value is either 10 Design Entry HDL coordinates or higher based on the grid size.

You can change this value. For this:

1. Select the *User Defined Threshold Value* option button.

   Note: Notice that a value of 0.017 is selected. This value equals 10 Design Entry HDL coordinates.

2. Enter a new value in *User Defined Threshold Value* check box or use the up- or down-arrow keys in the *User Defined Threshold Value* spin button to change the value.

You can also set the default threshold value for pin near wire end as grid-based. For this, select the *Grid Based Threshold Value* option button.

- For Decimal grids, the threshold value is grid size * 500.
- For Fractional grids, the threshold value is grid size * 400.
- For Metric grids, the threshold value is grid size * 20.

As an example, if you are using the Decimal grid type, and you specify a grid size of 0.01, Design Entry HDL will generate an error message when the distance between a wire end and a pin is less than or equal to 5 Design Entry HDL coordinates. This is because (0.01 * 500 = 5) is 5.

Arcs at Same Location

Checks for overlaid arcs.

Pins at Origin

Checks for pins at the origin (0,0) in symbol drawings.

Two Wires at Pins

Checks for wires overlapping a component at the pin.

Name Checks

Signal Names Checks for multiple names attached to the same signal.

Signal Names In Symbols Checks for the SIG_NAME property on a pin in a symbol file.

Miscellaneous Checks

Symbol Place Holders Checks for placeholder components that appear due to changes in the related library.

Property Place Holders Checks for placeholder properties that appear due to changes in the related library.

Multipackage Sections Checks for multiple SEC-type properties on an instance.

Page Number Mismatch Checks and corrects the PAGE_NUMBER directive conflicts in the ASCII and binary files for all the pages of the design.

Online Checks

Wire Short Check During Move If you select this option, Design Entry HDL displays the following error message when nets are shorted while moving components and nets in a design:

```
This action has resulted in change in
connectivity. Use undo to revert the
changes.
```

Design Entry HDL Options-Output

Command Related Info

Use these setup options to establish Design Entry HDL editor default settings for saving drawings.

Binary File Saves a binary representation of the logic.

Remove ASCII File

Removes the existing ASCII files in your schematic, when you want only binary files to be written. This option is enabled if you direct Design Entry HDL to save only the binary files when the schematic is written, by selecting the *Binary File* check box.

If this option is selected, the `.csb` file(s) is saved and the `.csa` file(s), if present, is deleted. This option works in accordance with the schematic write operation. If you save only the current page with this option on, only the ASCII file for the current page is deleted. Otherwise, the ASCII files for all the pages of the design are deleted.

ASCII File

Saves an ASCII representation of the logic.

Remove Binary File

Removes the existing binary files in your schematic, when you want only ASCII files to be written. This option is enabled if you direct Design Entry HDL to save only the ASCII files when the schematic is written, by selecting the *ASCII File* check box.

If this option is selected, the `.csa` file(s) is saved and the `.csb` file(s), if present, is deleted. This option works in accordance with the schematic write operation. If you save only the current page with this option on, only the binary file for the current page is deleted. Otherwise, the binary files for all the pages of the design are deleted.

If you choose to save both binary and ASCII files by selecting the *Binary File* and *ASCII File* options, the *Remove ASCII File* and *Remove Binary File* options will be disabled.

Confirm Write

Provides confirmation about saving the drawing.

Dependency File

Saves an ASCII file with dependency information.

Allowed Global Shorts	Allows you to add the list of global signals that you want to remain shorted in the design. If two global signals are shorted in your design, errors are displayed when you save or package the design. When you save a design in Design Entry HDL, error messages are displayed only for the global signals that are shorted within the block you are currently editing. When you package the design, error messages are displayed for global signals that are shorted within and across all the blocks in the design. If you want to allow the global signals to be shorted, type the name of first global signal in the **Signal1** field and the name of the second global signal in the **Signal2** field. Error messages are not displayed if the global signals listed in the **Allowed Global Shorts** list are shorted. For more information, see Shorting of Global Signals.

Design Entry HDL Options-General

Command Related Info

Use these setup options to establish Design Entry HDL editor default settings for paths, preferences, and message display.

Preferences	
Save Layout at Exit	Saves window and toolbar settings when you exit Design Entry HDL.
Click to Activate View	Activates a window when you click in it. Otherwise, a window is activated when you move the cursor into it.
Cursor Shapes	Enables different cursor shapes based on command mode.

Window Autopan

Enables panning behavior that lets you move the window over the drawing, rather than move the drawing inside the window.

Ctrl + RMB Context Menu

Changes the behavior of the right mouse button (RMB).

If the option is turned off:

Clicking right displays the context (pop-up) menu and

Pressing Ctrl+RMB causes a command-dependent action.

If the option is turned on, this functionality is reversed, where clicking right causes a command-dependent action and pressing Ctrl+RMB displays the context menu.

Ctrl+LMB Select and Drag

Changes the behavior of the select and drag mouse operation and for running commands with strokes.

If the option is turned off, do one of the following:

- Select an object on the schematic and drag the mouse to move the object.
- Press the left mouse button and drag the mouse to select multiple objects on the schematic. To exclude components, properties or wires from the selected objects, right-click and choose **Exclude** to exclude components, properties or wires from the selected objects. You can now click on one of the selected objects and drag the mouse to move all the selected objects.
- Press **Ctrl** or **SHIFT** and hold down the left mouse button to run commands with strokes.

If the option is turned on, do one of the following:

- Press **Ctrl**, select an object on the schematic and drag the mouse to move the object.
- Press **Ctrl** or **SHIFT**, hold down the left mouse button and drag the mouse to select multiple objects on the schematic. To exclude components, properties or wires from the selected objects, click the right mouse button and choose Exclude to exclude components, properties or wires from the selected objects. You can now press **Ctrl**, click on one of the selected objects and drag the mouse to move all the selected objects.
- Hold down the left mouse button to run commands with strokes.

Retain Version on Replace

When you select this check box and replace a symbol with another symbol on the schematic, the version of the existing symbol on the schematic is retained.

Show Physical Net Name

When you select this check box, DE-HDL displays the physical net names of signals and buses in the schematic. This is particularly useful when you want to view the winning hierarchical net name in a hierarchical design.

Tip

When Show Physical Net Name is selected, signal names cannot be added to wires. As a result, to add net names, you need to deselect the Show Physical Net Name option then select it again to view the physical net name.

To avoid toggling back and forth, you can use *Tools — Customize — Commands* and create shortcuts for the following commands:

- ❑ `set show_pnn_signame on`
- ❑ `set show_pnn_signame off`

Next, create a shortcut using *Tools — Customize — Keys*. You can then store the shortcut keys in `concepthdl_menu.txt`, stored, by default, in the `HOME\cdssetup\concept` directory.

Retain Location on Replace

If you select this option, when replacing or copying a component to a schematic, the soft location property ($LOCATION) of the component is retained, that is, the reference designator.

Component Browser (Add)

Opens the **Add Component** dialog box when you enter the add command in the console window and then press **Return**.

Show Category View (Add)

Displays the **Category View** page by default when you open the **Add Component** dialog box or when you enter the add command in the console window and then press **Return**.

If this check box is not selected, the **Library View** page is displayed by default when you open the **Add Component** dialog box.

Drawing Browser (Edit)

Activates the **View Open** dialog box when you enter the edit command in the console window and then press **Return**.

Libraries Browser (Lib)

Activates the **Search Stack** dialog box when you enter the lib command in the console window and then press **Return**. If off, the current search stack is displayed.

Show PPT Browser

Automatically opens the **Physical Part Filter** dialog box when you open the **Add Component** dialog box or when you enter the add command in the console window and then press **Return**.

Enable Pre-Select Mode

Activates the pre-select mode for Design Entry HDL menus.

Pre-select mode is not supported for console commands.

Set PATH property invisible

When you instantiate a component, the value of its `PATH` property is visible by default. By selecting this option, you can hide the `PATH` property of components when they are instantiated. The visibility of the existing components is not affected by selecting or deselecting this option.

Retain Hard Location on Replace

Select this checkbox if you want the hard location properties of a component to be retained when the component is being replaced by another on the schematic.

Messages

Specifies where you want certain types of messages displayed. These are messages that do not require any input from the user.

For example, when you set a very small (0.002) logic grid size and pan the drawing, Design Entry HDL gives the following warning “ Grid too small. Not displayed”. This is displayed in the console window or in a dialog box depending upon the option you select here.

If you select **Command Line**, Design Entry HDL displays the messages in the Console Command Window. If you select **Dialog**, Design Entry HDL displays the messages in a dialog box. If you select **Suppress**, Design Entry HDL does not display the type of messages.

Canonical Names

Allows you to control the display of canonical names in the Global Find, Global Navigation and **Attributes** dialog boxes by selecting or deselecting the **Library**, **Cell** or **View** check boxes.

The canonical name for a component is displayed in the following format:

@library.cell[view]:pagenumber_<value of path property on component>

The canonical name for a signal is displayed in the following format:

@library.cell[view]:pagenumber_<signal name>

If you want to view only the cell name in the canonical name, select the **Cell** check box and deselect the **Library** and **View** check boxes. The canonical name for a component will be displayed in the following format in the **Global Navigation window,** and the **Global Find** and **Attributes** dialog boxes:

.cell:pagenumber_<value of path property on component>

If you deselect the **Library**, **Cell** and **View** check boxes, the canonical name for a component will be displayed in the following format in the **Global Navigation window,** and the **Global Find** and **Attributes** dialog boxes:

:pagenumber_<value of path property on component>

Depth

Specifies the levels of Lib.Cell:View that is shown in a canonical name.

Page Border

Specifies a default page border for all new pages. Specify the name of the page border in the **Symbol** field. Specify the version of the page border symbol in the **Version** field. You can also click on the **Browse** button to select a default page border from the project libraries.

Drawings

Maximum Drawings	Specifies the maximum number of viewports that you can open in a session of Design Entry HDL. The default value is 50.

Design Entry HDL Options-Font

Command

Use the Font dialog box to specify font attributes for different types of schematic text objects.

UI Option	**Description**
Enable Font Support	Select this check box to enable support for fonts in DEHDL. This option is selected by default when you open any design in 16.3.
Category	Select the schematic text object for which you want to set the font and font attributes. You can set font attributes for different categories of text objects simultaneously.
Name	Select a font name to display a specific category of text objects. For example, you can select the Courier font to display all the net names in the design. The Name drop-down list includes all the fonts installed on the local system.

UI Option	Description
Size	Specify a font size with which all the newly added text objects for the category are to be displayed. This size is also known as point size, where one point size equals 1/72 of an inch. Currently, the font size stored in the DE-HDL database is a ratio of size in inches to 0.082. Therefore, 1 inch font size is written in the database as 1/0.082 = 12.095. With Font Support in DE-HDL, the sizes being displayed are point sizes, where 72 points = 1inch. Therefore, all the text sizes which are currently available in the database are converted to the point size and displayed accordingly. **Note:** The text size is currently stored in the database. Therefore, you can specify the font size for individual objects. All the objects which are already on the canvas have a font size specified on them and the same font size is honored.
Style	Select a font style from the four font styles: Regular, Bold, Bold Italic, and Italic. All fonts do not support all the styles. Therefore, you can specify only those styles which are supported for a specific font. For example, you can specify all the four styles for the Arial font, while only Regular style is supported for Arial Black.
Color	Select a color from the following list of colors with which all the newly added text for the category is to be displayed: Red, Blue, Green, Yellow, Orange, Salmon, Violet, Brown, Sky blue, White, Peach, Pink, Purple, Aqua, Gray, Mono. **Note:** The text color, like text size, is currently stored in the database. Therefore, you can specify the font color for individual objects. All the objects which are already on the canvas have a font color specified on them and the same font color is honored.
Effects	Select the Underline effect to display the text as underlined. By default, all text objects display regular text.
Reset	Resets the font settings back to the Cadence default.

UI Option	Description
Actual Preview	Shows the preview of sample text with the selected fonts and font attributes in the actual size it will appear on schematic canvas. If this check box is not selected, the sample text appears in a default size.
Details	Displays information about the selected font attribute.

Design Entry HDL Options-Split Symbol

Use the Split Symbol dialog box to specify settings for creating split symbols manually or using the auto-distribution method.

UI Option	Description
Warning on Partial Instantiation	If this option is selected, a warning message appears during save or check error operation when a hierarchical split symbol is partially instantiated, that is all the split symbols are not instantiated in the design.
Split Vector Ports	If this options is selected all the vector ports are bundled into a bus. As a result, the bus can be placed on the same hierarchical split symbol during port distribution.
Generate First Symbol as Full Symbol	If this option is selected, the first split symbol that is generate is the full symbol. If this option is not checked, symbol 1 is generated as part of the hierarchical split symbol.
Auto Distribution with	Provides four methods of auto-distribution for split symbols.
Fixed Number of Ports	Use this method when split symbols have to be created based on the number of ports on each symbol. The number of ports on each symbol is determined by the port count you specify in the *Ports Per Symbol* field.
Ports per Symbol	The number of ports allowed on each split symbol.
Ports on Same Page	Use this method to place all the ports on the same schematic sheet on the same symbol.

UI Option	Description
Pattern Based	Use this method to distribute ports based on the port name pattern specified in the grid. Each row of the grid generates a separate hierarchical split symbol based on the defined pattern. Multiple patterns can be specified separated by a comma. For example, In* or Pow*. You need to click the + sign to add rows to the grid and in each row specify the pattern to be used.
Property Based	Use this auto-distribution method to distribute ports based on the defined property value. Ports with the same property value are placed on the same symbol.

Design Entry HDL Options-Plotting

Procedure Command Related Info

Use this dialog box to specify Design Entry HDL default settings for plotting. Design Entry HDL supports the normal Windows plotting function.

Windows	Select this to use the Windows facility for plotting.
HPF	Select this to use the HPF facility. For more information, see HPF options.

Scaling	
Single Line Width	Adjusts the width of lines used to draw wires, boundary of components, and text.
Double Line Width	Adjusts the width of lines used to draw buses.
Adjust To __ % Normal Size	Increases or reduces the drawing size to the specified percentage.
Fit to Page	Adjusts the size of the drawing so that it fits into one page of the specified paper size.

Set Plot Margins to None

Clear this to set margins on sides of the paper for plotting the schematic.

By default, no margins are left on sides of the paper for plotting.

Plot Method

Screen Contents

Plots the portion of the schematic that is displayed on the screen.

Sheet Contents

Plots an entire page.

Color

Plots a drawing in color if you are using a color printer, or in gray scales if you are using a black and white printer.

Black and White

Plots a drawing in black and white.

Plot Font

Specifies the font to be used when the schematic is plotted. You can specify `Arial`, `Helvetica`, `Verdana`, `Trebuchet MS`, or `Default`. If you do not specify any font, Design Entry HDL uses the `Default` font, which was available before the 15.0 release.

Note: To use these fonts in plots, you must have them installed on your machine.

Plotter

Setup

Specifies the plotter to use, paper size and orientation, number of copies, and other print properties.

Design Entry HDL Options-Signal Integrity

Command   Related Info

Use this page to specify pin models that you want to assign to pins by default.

Default IO Cell Models Specify default models for various pins in this group box. You can specify the default pin models for the following pin types:

- IN
- OUT
- BI
- TRI
- OCL
- OCA

Design Entry HDL Options-Metadata Options

Command Related Info

Use this page to configure Allegro Design Entry HDL to create metadata for a design project.

Generate Schematic Metadata Generates schematic-related metadata required by the Component Revision Manager in Allegro Design Entry HDL.

Launch Component Revision Manager on concept invocation Ensures that the schematic is automatically checked for differences between the library cells and schematic cells when you launch Allegro Design Entry HDL.

Launch Component Revision Manager on Page Edit Ensures that the schematic is automatically checked for differences between the library cells and schematic cells when you edit a page.

Design Entry HDL Options-Design Navigation

Related Info

Use to display page names along with the block names and page number ranges under each block in the design.

Hierarchy Viewer Options
The *Design Navigation* page contains Hierarchy Viewer options, which can be configured to change the look and feel of Hierarchy Viewer.

Hide Sheet Numbers: Shows or hides sheet numbers or sheet number range in Hierarchy Viewer.

Hide Instance Names: Shows or hides instance names for the lower-level blocks in a design in Hierarchy Viewer.

Show Hierarchy Pages: Shows or hides hierarchy page name for different pages in the design.

Property on page border for page name
Helps you configure the property name to be picked from the page border to obtain the page name. This option sets the property on the page border for page name. By default, the property name mentioned is **TITLE**. This property name is configured in the *site.cpm* file. Notice that the option buttons for the *For Page Name property value* are now enabled.

For Page Name property value
Helps you configure to view the value of the Page Name property in the same case, lower case, or upper case.

Plot to File Options

Use this dialog box to specify options for plotting a schematic to a file.

Location
Enter the location of the file in which you want to print the drawing. The default location is the project directory.

Single File
Select this check box if you want to print all the pages of the design in a single file.

Name
The default filename is `vw.spool`. You can also specify another name if you want.

File Per page
Select this check box if you want to print every page of the design in a separate file.

Prefix Specify the prefix for the filenames in which pages of the design will be printed. The default prefix is `vw.spool`.

Example:

If you want to print pages 5 to 9 of the design and the prefix for filename is `vw.spool`, the output files will be as below:

page 5 in `vw.spool`-1

page 6 in `vw.spool`-2

page 7in `vw.spool`-3

page 9 in `vw.spool`-4

page 10in `vw.spool`-5

Global Find

Procedure

Use this dialog box to

- Initiate a global search for a specified net or all instances of a specified cell.
- Review the search results in a list, which identifies the object instances by hierarchical name or library location.
- Locate each net or cell instance listed in the search results. You can zoom in on a selected search result, which is highlighted in the design.

Note: Global Find does not find symbols with HDL_POWER and COMMENT_BODY=TRUE properties.

Name Type the name of the net or part for which all instances will be found, or select a previously entered name from the drop-down list.

To search for a vectored signal DATA(3..0) or DATA[3..0], type DATA or DATA<3..0>.

Find Starts the global search.

Using Wild Card Select this to use wild card in the name.

You can search for objects using wildcards such as: ls*, *ttl*, etc.

Always use ‘*’ for a wildcard search.

Important

Wild cards are not supported in property name and value.

Net Select this to specify that you want to locate an object of this type.

Cell Select this to specify that you want to locate an object of this type. (This is selected by default)

With Properties

Name Optional. Type a property name to be searched.

Value Optional. Do one of the following:

Type the value of the property to be searched.

Enter an * (asterisk) to find all objects having the specified property name with any value.

Results

Hierarchical Names

Lists the results of the search by the full canonical name for each instance.

The canonical name for a component is displayed in the following format:

@library.cell[view]:pagenumber_<value of path property on component>

The canonical name for a signal is displayed in the following format:

@library.cell[view]:pagenumber_<signal name>

You can control the display of the canonical name by selecting or deselecting the **Library**, **Cell** or **View** check boxes in the **Canonical Names** group box in the General page of the **Design Entry HDL Options** dialog box.

For example, if you want to view only the cell name in the canonical name, select the **Cell** check box and deselect the **Library** and **View** check boxes. The canonical name for a component will be shown in the following format in the **Global Find** dialog box:

.cell:pagenumber_<value of path property on component>

If you deselect the **Library**, **Cell** and **View** check boxes, the canonical name for a component will be shown in the following format in the **Global Find** dialog box:

:pagenumber_<value of path property on component>

Library Locations

Presents the search results in spreadsheet format, with columns listing Object, Page, Library, Cell, and View information for each instance.

Status Area

When the global search for the specified object finishes, this unlabeled area lists the results of the global search for the specified net or cell.

Zoom to Object

Indicates the selected search result is to be viewed by zooming in on it.

Navigate

Displays the Global Navigation dialog box for reviewing connectivity across the design.

Clear

Clears the fields in this dialog box and restores the default settings.

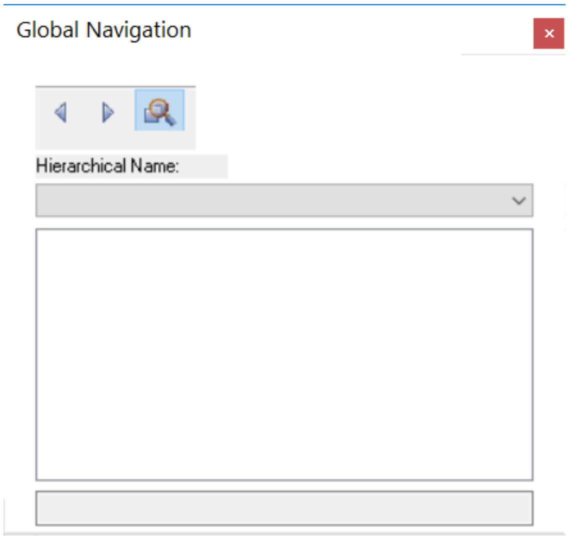
Close

Closes the **Global Find** dialog box.

Global Navigation

Procedure

Use this window to find and view all synonyms and aliases for a selected signal or cell (part) within a schematic design or when cross-probing between Design Entry HDL and PCB Editor.



Hierarchical Name Displays the full hierarchical path name of a net or cell that you select in your design.

Hierarchical Names

Lists the results of the search by the full canonical name for each instance.

The canonical name for a component is displayed in the following format:

@library.cell[view]:pagenumber_<value of path property on component>

The canonical name for a signal is displayed in the following format:

@library.cell[view]:pagenumber_<signal name>

You can control the display of the canonical name by selecting or deselecting the **Library**, **Cell** or **View** check boxes in the **Canonical Names** group box in the General page of the **Design Entry HDL Options** dialog box.

For example, if you want to view only the cell name in the canonical name, select the **Cell** check box and deselect the **Library** and **View** check boxes. The canonical name for a signal will be shown in the following format in the **Global Navigation** window:

.cell:pagenumber_<signal name>

If you deselect the **Library**, **Cell** and **View** check boxes, the canonical name for a signal will be shown in the following format in the Global Navigation window:

:pagenumber_<signal name>

Status Area

When the selected object is located in the design, this unlabeled area lists the results of the global search for the specified net or cell.

Back and Next buttons

Helps you navigate between different nets and part instances that you select through the Global Navigation window during a single session of Design Entry HDL

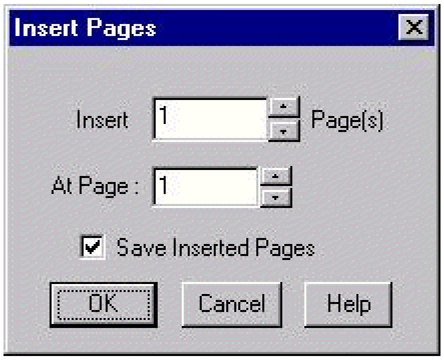
Zoom to Object

Indicates the selected instance is to be viewed by zooming in on it.

Insert Pages

Procedure Command

Use this dialog box to insert a page or a set of pages in a schematic.



Insert: 1
At Page : 1
☑ Save Inserted Pages

Field	Description
Insert	Indicates the number of pages you want to insert. The maximum number of pages that you can insert is 250. The default value of this field is 1.
At Page	Is the location from where you want to insert the page(s). Pages are always inserted before the current page at the target location.
Save Inserted Pages	Creates Page* files for the pages to be inserted. If this option is deselected, a page gap will be created in the schematic. Page gaps do not have corresponding Page* files.

For more information, refer to the Performing Page Management section of the *Allegro Design Entry HDL User Guide*.

Markers

Use this dialog box to access Markers controls.

File

Load

Lets you navigate to the marker file you want to load.

Save

Saves the current marker file.

Save As

Saves the current marker file with a new name.

Close

Closes the **Markers** control window.

Edit

Undo Delete

Undoes the deletion of the last deleted marker.

Only the last deleted marker can be undeleted. If you have not saved the markers file after deleting multiple markers and want to undo more than one delete, reload the markers file to start over. Markers are deleted only when the markers file is saved.

Delete Marker

Deletes markers from the **Messages** list.

View

As an alternative to using the **View** menu to navigate markers, you can use the toolbar in the **Markers** control window.

First Marker

Selects the first marker listed in the marker file.

Previous Marker

Selects the previous marker listed in the marker file.

Next Marker

Selects the next marker listed in the marker file.

Last Marker

Selects the last marker listed in the marker file.

Previous Location

Selects the previous location in the currently selected marker.

Next Location

Selects the next location in the currently selected marker.

Filter Options

Filters errors by

Rule name

Object type

Severity level

Expression

Details

Lists the long error message associated with short error messages that appear in the **Messages** list. You can also click **Details>>** to view details on error messages.

Toolbar
Displays the toolbar in the **Markers** control window. This is similar to the **Markers** toolbar you can display in Design Entry HDL.

Messages
Lists short error messages for each marker. Select a message to highlight the associated marker on the schematic. Click **Details>>** to view the long error message for a specified marker.

Location List
Lists error locations within markers. Select a location to highlight the error at the specified location.

A single marker can contain any number of locations. If you select an error from the **Messages** list that includes multiple objects, you can navigate to the next error location to highlight each object separately.

If there are three locations for a marker and the first location is selected in the Location List combo box, Design Entry HDL displays 1/3 near the Location List label. If you select the second location, Design Entry HDL displays 2/3. (The numerator represents the selected location and the denominator represents the total number of locations for the selected error.

Details>>
Lists the long error message associated with short error messages that appear in the **Messages** list.

Markers: Filter

Use this dialog box to specify filter options for markers.

Selection By Rule Name
Filters markers by the specified rule set.

Object Type
Filters markers by the specified object type associated with marker location.

Severity
Filters markers by error severity: **Info**, **Warning**, **Error**, and **Oversight**.

Message String
Filters markers by the specified string.

Exclude	Omits the specified message string from the filter.
Select All	Selects all markers.
Deselect All	Filters out all markers.
OK	Filters markers based on your selections.
Cancel	Cancels your selections and closes the dialog box.

Genview

Procedure Command

Genview lets you generate a design view from an existing view or file. A design can be represented by the following views:

- Schematic (SCH)
- Symbol (SYM)
- VHDL
- Verilog

Source

Lib.Cell:View

Specify the view to be used as the source in the following format:

`lib.cell:view`

By default, the view for the current drawing is displayed as the source view.

Browse

Displays the **View Open** dialog box. Select the library, cell and view and click **Open**.

File

Specify the name of the VHDL or Verilog text file that you want to use to generate the design view. *Tools >Generate View* extracts the design name from the entity name (VHDL) or module name (Verilog) in the source file and creates a cell in the destination library.

The module names and the pin names in Verilog source files must be in lowercase only.

When Genview creates a schematic, it creates a single page schematic with only the interface ports.

Browse

Displays the **Specify HDL file** dialog box. Select the source Verilog or VHDL file and click **Open**.

Type

Specify whether the source file is Verilog or VHDL.

This field is enabled only if the **File** option is selected.

Destination

Library

Select the library where you want Design Entry HDL to create the destination cell if the source is a file.

If the source is a view, the destination library is the same as the library for the source view.

View

Select the view that you want to generate.

Type

Select the type of the view you have selected in the View field as one of the following:

Select **Schematic** if you have selected the **sch_1** view.

Select **Symbol** if you have selected the **sym_1** view.

Select **VHDL** if you have selected the **vhdl_1** view.

Select **Verilog** if you have selected the **vlog_1** view.

Retain Graphics

Select this check box if you want to retain the placement of pins that already existed on the graphic for the symbol.

For example, suppose that the symbol view already exists. If you add or delete a pin in the source view or source file and regenerate the symbol view, the placement of the pins that already existed (pins that were not deleted in the source view or source file) on the symbol will be retained.

Cadence recommends that you use this option if you have already used the symbol on your schematic. This will ensure that the connectivity between a wire and a pin of the symbol on the schematic is not lost because the placement of the pin on the symbol does not change.

If you do not select this check box, the graphic for the symbol is regenerated and the pin placement is done by Design Entry HDL using its internal algorithms.

See Retain Graphics and Split Vectored Ports Example

Split Vector Ports

Select this check box if you want the vectored ports in the source view or source file to be split into multiple pins (representing each bit of the vectored port) on the symbol.

For example, if the source view or source file has a vectored port DATA<3..0>, the following four pins will be added on the symbol:

DATA<3>

DATA<2>

DATA<1>

DATA<0>

If this check box is not selected, the symbol will have a pin named DATA<3..0>.

See Retain Graphics and Split Vectored Ports Example

Split Symbols

Enables you to perform the settings required for generating a *hierarchical split symbol*.

------Setup

Opens the Design Entry HDL Options-Split Symbol dialog.

------Distribute Ports

Opens the *Distribute Port* dialog.

Generate

Click to generate the specified view.

Done	Click to close the dialog box.
Output	Displays the results of the generate view process.

The Verilog or VHDL file, whichever is specified, shall always be imported.

For a Verilog file, every module is imported under the vlog_rtl directory.

The VHDL file that Genview generates from a symbol view contains the following text:

```
architecture <name_of_architecture> of <name_of_entity>
     begin
     end <name_of_architecture>
```

Example

```
architecture <abc> of <ent>
     begin
     end <abc>
```

A VHDL file is imported by placing the various sections of the file in separate vhdl.vhd files. The sections are as follows:

Section a - <library/use clauses #0>
Section b - <package ttt is>
Section c - < library/use clauses #1>
Section d - <package body ttt is>
Section e - <library/use clauses #2>
Section f - <entity xxx is>
Section g - < library/use clauses #3>
Section h - <architecture yyy of xxx is>

Where sections a, b, c, d, e, and g are optional.

The `vhdl.vhd` files are placed in the following directory structure:

xxx/entity/vhdl.vhd containing sections e and f
xxx/yyy/vhdl.vhd containing sections g and h
ttt/package/vhdl.vhd containing sections a and b
ttt/body/vhdl.vhd containing sections c and d

So, the entity name determines the cell name, and the architecture name determines the view within the cell. An entity section can have multiple architecture sections associated with it. These form multiple views within the cell for the entity. The file can have multiple entity sections also implying multiple cells.

This can be understood through the following example:

The VHDL file being imported is:

```
library lib;
package abc is
end abc;


library lib;
package body abc is
end abc;

------------------------------------
-- ENTITY DECLARATION
------------------------------------
library lib;

use LIB.STD_LOGIC_1164.ALL;
use WORK.ALL;

entity ent is
    port (ADDRESS:OUT STD_ULOGIC_VECTOR (15 DOWNTO 0);
    DATAOUT:OUT STD_ULOGIC_VECTOR (15 DOWNTO 0);
    STALL:IN STD_ULOGIC   );
end ent;

library trial;
------------------------------------
-- ARCHITECTURE DECLARATION
------------------------------------
architecture schematic  of ENT is
    component REG16
        port (
            REGIN : IN STD_ULOGIC_VECTOR(15 DOWNTO 0);
            REGOUT : OUT STD_ULOGIC_VECTOR(15 DOWNTO 0);
            CLK  : IN STD_ULOGIC;
            SDO  : OUT STD_ULOGIC
        );
    end component;

    component CLOCKGEN

        port (
            CK   : OUT STD_ULOGIC;
            RD   : OUT STD_ULOGIC;
            WR   : OUT STD_ULOGIC;
            CLOCK : IN STD_ULOGIC;
            STALL : IN STD_ULOGIC;
            IR2ISSTORE : IN STD_ULOGIC;
            RESET : IN STD_ULOGIC
        );
    end component;
END schematic;
```

The directory structure formed is:

- ent/entity/vhdl.vhd containing

```
library lib;
    use LIB.STD_LOGIC_1164.ALL;
    use WORK.ALL;
    entity ent is
        port (ADDRESS:OUT STD_ULOGIC_VECTOR (15 DOWNTO 0);
        DATAOUT:OUT STD_ULOGIC_VECTOR (15 DOWNTO 0);
        STALL:IN STD_ULOGIC   );
    end ent;
```

- ent/schematic/vhdl.vhd containing

```
library trial;
architecture schematic  of ENT is
component REG16
        port (
            REGIN : IN STD_ULOGIC_VECTOR(15 DOWNTO 0);
            REGOUT   : OUT STD_ULOGIC_VECTOR(15 DOWNTO 0);
            CLK  : IN STD_ULOGIC;
            SDO  : OUT STD_ULOGIC
        );
    end component;
component CLOCKGEN
        port (
            CK   : OUT STD_ULOGIC;
            RD   : OUT STD_ULOGIC;
            WR   : OUT STD_ULOGIC;
            CLOCK : IN STD_ULOGIC;
            STALL : IN STD_ULOGIC;
            IR2ISSTORE : IN STD_ULOGIC;
            RESET : IN STD_ULOGIC
        );
    end component;
END schematic;
```

- abc/package/vhdl.vhd containing

```
library lib;
    package abc is
    end abc;
```

- abc/body/vhdl.vhd containing

```
library lib;
    package body abc is
    end abc;
```

Batch Commands

For generating a view from a source view:

```
genviewHDL -r lib.cell:view -o <output view> -vtype VHDL_Verilog_Symbol_Schematic
     [-temp <path to temp directory>] [-overwrite] [-splitports]
```

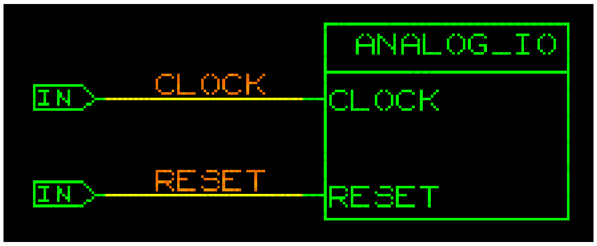
For generating a view from a source Verilog or VHDL file:

```
genviewHDL -r <destination library> -file <file path> -ftype VHDL_Verilog -o
     <output view> -vtype VHDL_Verilog_Symbol_Schematic [-temp <path to temp
     directory>] [-overwrite] [-splitports]
```

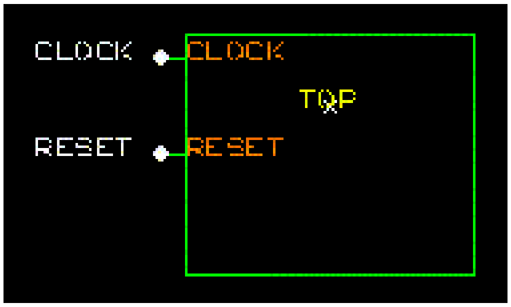
Retain Graphics and Split Vectored Ports Example

The Retain Graphics and Split Vector Ports options are explained below using an example.

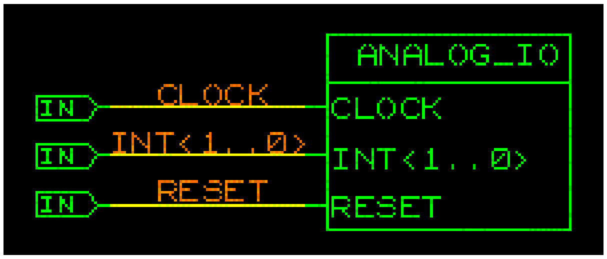
Suppose that you have a schematic `TOP.SCH.1.1` as below:



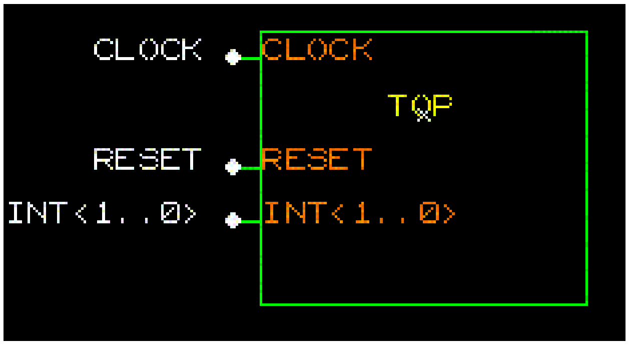
1. Generate the symbol for the schematic. A symbol named `TOP` will be created as below:



2. Add a pin `INT<1..0>` on the `ANALOG_IO` block and connect it to an input port `INT<1..0>` on the schematic, as below:



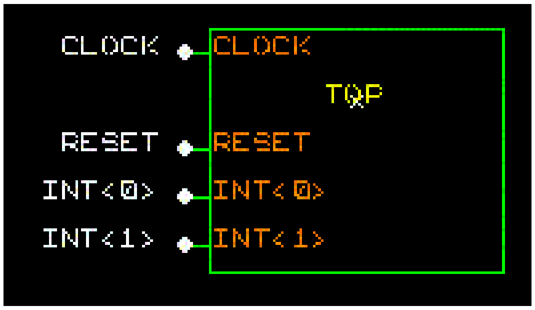
3. Generate the symbol for the schematic again with the *Retain Graphics* check box selected. The symbol `TOP` will be created as below:



Note that the placement of the pins `CLOCK` and `RESET` on the symbol have not changed.

Suppose that you have instantiated the symbol `TOP` on some other schematic page and have connected a wire to the pin `CLOCK` on the symbol. The connectivity between the wire and the pin `CLOCK` is not lost now because the placement of the pin on the symbol has not changed.

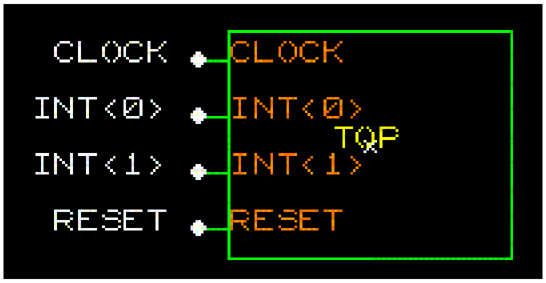
4. Generate the symbol for the schematic again with the *Retain Graphics* and *Split Vector Ports* check boxes selected. The symbol `TOP` will be created as below:



Note that the vectored port `INT<1..0>` in the schematic has been split into two pins (representing each bit of the vectored port) `INT<0>` and `INT<1>` on the symbol. The vectored pin `INT<1..0>` is deleted from the symbol, and the pins `INT<0>` and `INT<1>` are added as new pins on the symbol.

Also note that the placement of the pins `CLOCK` and `RESET` on the symbol have not changed.

5. Generate the symbol for the schematic again with the *Retain Graphics* check box deselected and the *Split Vector Ports* check box selected. The symbol `TOP` will be created as below:



Note that the placement of the pins on the symbol have changed. This is because the graphic for the symbol is regenerated when the *Retain Graphics* check box is not selected. The placement of the pins on the symbol is done by Design Entry HDL using its internal algorithms.

Suppose that you have instantiated the symbol `TOP` on some other schematic page and connected a wire to the pin `CLOCK` on the symbol. The connectivity between the wire and the pin `CLOCK` is lost now because the placement of the pin on the symbol has changed.

Relational Operators for Numeric Filtering

Use this operator	To perform this operation
=	Equals
!=	not equals
>	greater than
<	less than
>=	greater than or equal to
<=	less than or equal to
!	logical NOT
&	logical AND
|	logical OR
-	Range
( )	Parentheses to indicate the order of operations on an expression

Use this operator	To perform this operation
“ “ or ‘ ‘	double or single quotation marks (used to embed white space or operator characters in a string)

Command-Dependent Mouse Button Operation

In certain command modes, you can use the mouse to:

- Attach items at the nearest object
- Start wires at the nearest attachment point
- Select the nearest object when using Group commands
- Cycle through versions of components when using Part Information Manager

Basic Attributes Dialog Box

File Lets you load and save predefined templates of attributes and their values, alignment, and visibility properties.

File – Load Attributes adds new properties in the template to the property list but does not replace existing properties with the same name.

Object Displays the name of the object that you selected.

Name Lists the property name for a selected object. Default properties appear with a gray background and cannot be changed or deleted.

Value Lists the property value.

Visible Specifies the amount of property information displayed on the drawing: **None**, **Value** only, **Name** only, or **Both**.

Align Specifies the text alignment: **Left**, **Center**, or **Right**.

OK Closes the **Attributes** dialog box and saves changes.

Cancel Closes the **Attributes** dialog box without saving changes.

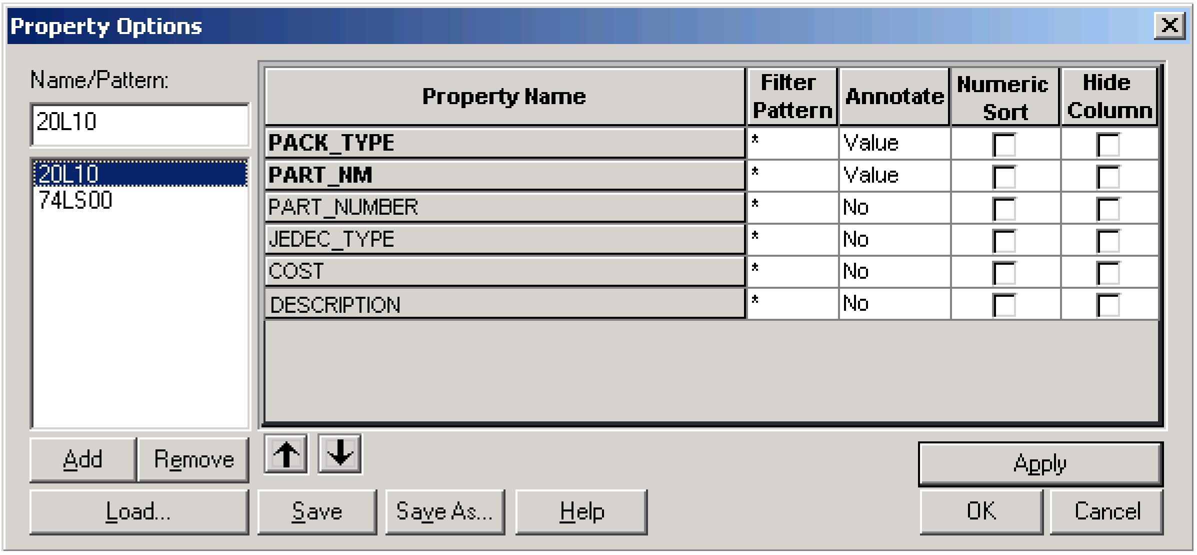
Add Adds a row to the listing in the dialog box so that you can enter a new property and value.

Delete Deletes a property. This button is disabled for properties that cannot be deleted. Clicking **Delete** removes the property from the listing in the dialog box. Clicking **OK** removes the property from the drawing.

Property Options

Use this dialog box to specify the property options that defines the format and visibility of the physical properties of a part. The Property Options dialog box appears when you right-click on a PPT row (*Search Results* pane of the Part Information Manager window) and choose *PPT Options* from the pop-up menu. Alternatively, click *PPT Options* in the Modify Component dialog box.

Figure 8-1 The Property Options dialog box



Name/Pattern: 20L10

20L10
74LS00

Property Name	Filter Pattern	Annotate	Numeric Sort	Hide Column
PACK_TYPE	*	Value	□	□
PART_NM	*	Value	□	□
PART_NUMBER	*	No	□	□
JEDEC_TYPE	*	No	□	□
COST	*	No	□	□
DESCRIPTION	*	No	□	□

Name/Pattern Contains the name of the currently selected option set.

Right pane (Option Set Properties pane) Contains the current list of properties for the option set you select in the *Name/Pattern* field.

Property Name The name of the key and injected properties appear under this column. Properties in bold are the key properties.

Filter Pattern

Lets you filter physical property values based on the string you enter.

For example, to filter the row of the PPT that has the value of VOLTAGE as 63V, select VOLTAGE property in the *Property Name* column, enter 63 in the *Filter Pattern* field, and click *Apply*.

Annotate

Controls the visibility of the physical property on the schematic.

If you select *No*, the physical properties do not appear on the schematic.

If you select *Name*, only the names of the physical properties appear on the schematic.

If you select *Value*, only the values of physical properties appear on the schematic.

If you select *Both*, the names as well as the values of physical properties appear on the schematic.

If you select *Invisible*, the physical properties are added on the schematic and are read by all tools, but they are not visible.

Numeric Sort

There are two types of sorting that Design Entry HDL employs to list the rows in a PPT file.

Numeric Sort

In numeric sorting, the lesser numbers are placed initially followed by the greater numbers (50<150).

String Sort

In string sorting, the first characters of two values are taken and compared. If they are the same, then the second characters are compared. (150<50 150 is less than 50 because 1 and 5 are compared together).

Hide Column

Suppress the appearance of the column in the *Search Results* pane (*Part Information Manager* window).

Add

Adds a new option set with the name that appears in the *Name/Pattern* field. If the name already exists, you have to add an option set with a new name.

Remove

Removes the current option set.

Load

Click this button to display a file browser that lets you load a completely new set of option sets.

If you specify the path to the PPT Option Set file in *Tools – Options – Paths*, the `ppt_optionset.dat` file will load, by default.

Save

Click to save the option sets you have defined using the Property Options dialog box to a file `(ppt_optionset.dat)` in the default location of the current project directory.

Save As

Click to save the option sets you have defined using the Property Options dialog box to a file (with `.dat` extension) other than the default file (`ppt_optionset.dat`).

↑

Lets you move the position of a property name up in the order. For example, for a physical part, if PART_NUMBER is the third cell of the *Property Name* column, and you select it and click the *Up* arrow twice. The property will move to the first cell. Now, the PART_NUMBER will be the first column that appears in a part table row.

↓

Lets you move the position of a property name down in the order. For example, for a physical part, if JEDEC_TYPE is the second cell of the *Property Name* column, and you select it and click the *Down* arrow twice. The property will move to the fourth cell. As a result, JEDEC_TYPE will be the fourth property column that appears in a part table row.

Apply

Reflects the changes you made in the option set on the Property Options dialog box and other related interface elements (such as Part Information Manager).

OK

Applies the changes in the *Search Results* pane (*Part Information Manager* window) and closes the Property Options dialog box.

Cancel

Closes the Property Options dialog box without saving the changes you made in the current session.

Help

Lets you view the online Help for the Property Options dialog box.

Add Part

Procedure

When you choose a component in Part Information Manager and want to attach physical properties to it, this dialog box appears when there is no part table file for the selected logical component. The *Add Part* dialog box reads the `chips.prt` file and lists the available `PACK_TYPE` property values. You can select a pack type and annotate it on the schematic.

Pack type Select the desired pack type.

Annotate You can set **No**, **Name**, **Value**, **Both**, and **Invisible**.

OK Click *OK* to add the part with only `PACK_TYPE` property.

Design Entry HDL Options-Paths

Command Related Info

Use this dialog box to set the default paths for various files used by the Design Entry HDL schematic editor.

Input Paths Specify the default paths for Design Entry HDL to read the following files.

Category File Path Specify the directory that contains the category (`.cat`) files used to organize components by category. The component categories are displayed in the **Category View** page of the Add Component-Category View dialog box.

Input Script Specify the path to the file that contains Design Entry HDL console commands to be run when you start Design Entry HDL.

PPT Option Set	Specify the path to the PPT Option Set file that you want Design Entry HDL to use by default. This file stores the default display settings for physical properties in the schematic and in the Physical Part Filter dialog box. For more information on defining physical property options, see the *Working with Libraries and Components* chapter of the *Allegro Design Entry HDL User Guide*.
Attribute Directory	Specify the path to the attribute (`.att`) file to be loaded in the Attributes dialog box. The default path is <*your_inst_dir*>/`tools/fet/concept/attributes`.

Toolbar Name

Enter the name of the new toolbar you wish to create.

Attribute Details

Use this dialog box to view details of the object you selected to display the *Attributes* dialog box, and the property you selected in the *Attributes* dialog box.

Value	Specifies the value of the property on the selected object.
Location	Displays the location of the object.
Value Type	Displays whether the property displayed in the Value field is Schematic or Occurrence.
Property Type	Displays whether the property is soft or hard.
Close	Closes the *Attribute Details* dialog box.

Strokes

Procedure Commands

Use this dialog box to define a stroke command. When you open this dialog box, it loads the default stroke definition file located at `<`*`your_install_dir`*`>/tools/fet/concept/concept.strokes`.

Define	Click to define a new stroke command. This button is enabled only if you change the existing console command in the field at the bottom of this dialog box.
Load	Click this button to load a stroke definition file. By default, Design Entry HDL loads the default stroke definition file `concept.strokes` located at `<`*`your_install_dir`*`>/tools/fet/concept/`.
Save	Click to save the stroke definition file.
Previous	Displays the previous stroke definition in the file.
Next	Displays the next stroke definition in the file.

To define a new stroke

1. Enter the console command that you want to execute using the new stroke in the field at the bottom of the dialog box.
2. Draw a stroke pattern in the space provided for defining strokes.
3. Click *Define*.
4. Click Save to save the file in your local project area.
5. Click *OK*.

Commands

- Loadstrokes
- Strokefile

Plot

Procedure Command

Note: If you want to plot occurrence properties, you must change to the Occurrence Edit mode before plotting the schematic.

Name

Select the printer you wish to use.

The field displays the default printer.

Print to file

Select the check box to plot the drawing to a file. Specify the name of the file in the *Print to File* dialog box that appears.

Plot Range

Select the Range of Plotting

All

Choose *All* to plot all pages in the design.

Pages From To

Choose *Pages From To* to plot specified pages.

Active Page

Choose *Active Page* to plot the currently open drawing.

Hierarchy

Choose *Hierarchy*, if you want to perform hierarchical plotting. Design Entry HDL extends the *Plot* dialog box to display the hierarchical structure of the root design.

Click + next to a design to expand the hierarchy tree of the design. The - icon indicates that the hierarchy tree of the design is expanded.

Click - next to a design to collapse the hierarchy tree of the design.

The root design is displayed in the format <*design_name*> (*page_number*). The *page_number* variable indicates the page number on which the design will be plotted. For example, if the root design name is `laptop` and has 1 page, the design name will be displayed as `laptop (1)`. If the laptop design has three pages, the design name will be displayed as laptop (1-3). Sub designs are displayed in the format <*design_name*> <*instance_identifier*> (*page_number*). For example, if the sub design is displayed as flashcard <`page1_i11`> (2), flashcard is the name of the sub design, <`page1_i11`> indicates that the sub design is instantiated as instance `i11` on page 1 of the root design, and (2) indicates the number of the page in which the sub design will be plotted.

You can select or deselect sub-designs for plotting. Select the check box next to a design to plot the design. If a sub design is deselected for plotting, the check box next to the parent design is displayed in a darker color tickmark sign.

Clear the check box next to a design if you do not want to plot the design.

If you choose *Hierarchy*, all the fields in the *Design* group box are disabled.

Clear All

Click to de-select all the designs that you have selected in the hierarchical design for plotting.

Plot

Click to plot the design.

HPF Plot

Procedure Commands

The default values are displayed for the currently open drawing.

Note: If you want to plot occurrence properties, you must change to the Occurrence Edit mode before plotting the schematic.

Design

Library

Displays the name of the library in which the currently open drawing exists. Change the library name, if required.

If you change the library name, ensure that the library is defined in the `cds.lib` file.

This field is disabled if you are in Occurrence Edit mode. In Occurrence Edit mode you can plot only the currently active drawing or drawings from the same version of the view in which the drawing exists.

Cell

Displays the name of the cell in which the currently open drawing exists. Change the cell name, if required.

If you change the cell name, ensure that the cell is present in the library you have specified in the *Library* field. You can use wildcards (* and ?) in this field.

This field is disabled if you are in Occurrence Edit mode. In Occurrence Edit mode you can plot only the currently active drawing or drawings from the same version of the view in which the drawing exists.

View

Displays the name of the cell in which the currently open drawing exists. Change the view name, if required.

Enter SCHEMATIC, if you want to plot schematic drawings.

Enter SYM, if you want to plot symbol drawings.

Enter SCHCREF_1, if you want to plot want to plot the schematic drawings generated by CRefer in the schcref_1 view.

If you change the view name, ensure that the view is present in the cell you have specified in the Cell field. You can use wildcards (* and ?) in this field.

This field is disabled if you are in Occurrence Edit mode. In Occurrence Edit mode you can plot only the currently active drawing or drawings from the same version of the view in which the drawing exists.

Version

Displays the version of the view in which the currently open drawing exists. Change the version number in the, if required.

The version number indicates the version of the view you want to plot. For example, if you want to plot the schematic drawings in the sch_1 view of a cell, specify SCHEMATIC in the *View* field and 1 in the *Version* field. If you want to plot the symbol drawings in the sym_3 view of a cell, specify SYM in the View field and 3 in the Version field. To plot the schematic drawings generated by CRefer in the schcref_1 view of a cell, specify SCHCREF_1 in the *View* field and 1 in the *Version* field.

By default, the version number is 1. If you change the version number, ensure that the version of the view is present in the cell you have specified in the *Cell* field. You can use wildcards (* and ?) in this field.

This field is disabled if you are in Occurrence Edit mode. In Occurrence Edit mode you can plot only the currently active drawing or drawings from the same version of the view in which the drawing exists.

Page

Displays the physical page number of the currently open drawing. Change the page number, if required.

By default, the page number is 1. If you change the page number, ensure that the page is present in the version of the view you have specified in the *Version* field. You can use wildcards (* and ?) in this field.

Plot Range Select the range for plotting.

Design Choose *Design* to plot the drawing specified in the *Design* group box.

Hierarchy Choose *Hierarchy*, if you want to perform hierarchical plotting. Design Entry HDL extends the HPF Plot dialog box to display the hierarchical structure of the root design.

Click + next to a design to expand the hierarchy tree of the design. The - icon indicates that the hierarchy tree of the design is expanded.

Click - next to a design to collapse the hierarchy tree of the design.

The root design is displayed in the format <*`design_name`*> (*`page_number`*). The `page_number` variable indicates the page number on which the design will be plotted. For example, if the root design name is laptop and has 1 page, the design name will be displayed as `laptop (1)`. If the laptop design has three pages, the design name will be displayed as `laptop (1-3)`. Sub designs are displayed in the format <*`design_name`*> <*`instance_identifier`*> (*`page_number`*). For example, if the sub design is displayed as `flashcard <page1_i11> (2)`, flashcard is the name of the sub design, `<page1_i11>` indicates that the sub design is instantiated as instance i11 on page 1 of the root design, and (2) indicates the number of the page in which the sub design will be plotted.

You can select or deselect sub-designs for plotting. Select the check box next to a design to plot the design. If a sub design is deselected for plotting, the check box next to the parent design is displayed in a darker color tickmark sign.

Clear the check box next to a design if you do not want to plot the design.

If you choose *Hierarchy*, all the fields in the *Design* group box are disabled.

Hierarchy

Note:

Clear All Click to de-select all the designs that you have selected in the hierarchical design for plotting.

Plot Click to plot the design.

Setup Click to display the Plot to File Options page of the *Design Entry HDL Options* dialog box.

Custom Text

Procedure

Use this dialog box to attach custom text to objects on a schematic. Within the custom text you add, you can use custom variables.

Format String Enter the format of the text you want to define.

For example, you can enter the following text in the *Format* field:

This is page

Variables Select custom variable(s) to add to the format string from the *Variables* drop-down list.

The Variables list contains the in-built Design Entry HDL variables and user-defined variables.

For example, you can select <CON_PAGE_NUM>

Display String The *Display String* displays the custom text after substituting the values of custom variables.

Alignment Select *Left*, *Center*, or *Right* depending on how you want the custom text to be aligned with the object.

Design Entry HDL Options-Custom Variables

Command Related Info

Use this dialog box to define Custom Variables for the current project. Custom Variables are variables you can define in Design Entry HDL for placing on a schematic. These variables make the plots of cross referenced schematics more illustrative and easy to use.

Name Enter the name of the Custom variable you wish to define.

Value Enter the value of the Custom variable.

> *Important*
>
> You cannot leave the value of the variable blank. The variable is deleted from the list if no value is specified.

Pattern

Use this dialog box to create a group by selecting objects that contain a common pattern of text. For example, if you enter *PORT, in the Pattern field and click OK, Design Entry HDL selects all objects in the current schematic that have PORT in their name and groups them.

Array Size

Use this dialog box to enter the number of instances of an object you want to create.

Scale Factor

Use this dialog box to zoom the current view of the design by a factor. For example, if you enter 2, the view is zoomed by 200%, and if you enter 0.5, the view becomes 50%.

Wire Pattern

Use this dialog box to specify the pattern for wires. The wires that you click on will have the pattern specified.

New Block Name

Use this dialog box to enter a new name for an existing block.

Group Name

Use this dialog box to set the name of the current group. The name can be any character from A to Z.

Component Name

Use this dialog box to enter the name of the component you want to replace all components of the group with. This component has to be available in the libraries being used in the design.

Enter New Command Name

Use this dialog box to enter the name of the new command you want to add.

Bus Name

Procedure Command

Use this dialog box to name a wire that has been broken into individual bits.

Bus Name	Enter the name of the bus.
MSB	Enter the bit number of the most significant bit.
LSB	Enter the bit number of the least significant bit
Increment	Enter the bit difference between two adjacent bits.

Bus Tap Range

Use this dialog box to enter the range of the tap bits of a bus.

MSB	Enter the bit number of the most significant bit.
LSB	Enter the bit number of the least significant bit
Increment	Enter the bit difference between two adjacent bits.

Property

Procedure Command

Use this dialog box to attach new properties to an object in the schematic.

Property Name	Enter the name of the property
Property Value	Enter the value of the property

Group Contents

Procedure

Use this dialog box to view the objects that form the current group.

Contents	Shows the objects available in the current group
Show	Highlights all the objects in the current group.
Clear	Clears the highlighting that occurs when Show button is pressed.
Close	Closes the dialog box.

Go To Page/Symbol

Use this dialog box to enter a page number for

- editing a page of a multi-page schematic
- editing a version of a symbol

- creating a new page in a schematic

Page/Symbol	■ If you are editing a schematic, enter the page you want to edit. If the **Calculate page number in hierarchy** option is not selected, the page number is calculated with reference to the current level of hierarchy. If the **Calculate page number in hierarchy** option is selected, the page number is calculated with reference to the entire design. ■ If you are currently editing a version of a symbol, enter the new version you want to edit. ■ If the page (or version of symbol) that you have entered does not exist in the current level of hierarchy, you are prompted to create a new page (or version of symbol).
Calculate page number in hierarchy	Select to calculate the **Page/Symbol** number with reference to the entire design (and not just the current level of hierarchy). For details on page numbers in a hierarchical design, refer the CURRENT_DESIGN_SHEET and TOTAL_DESIGN_SHEETS Custom Text Variables section of the Allegro Design Entry HDL User Guide.

Save Files

Use this dialog to save your files before you exit Design Entry HDL.

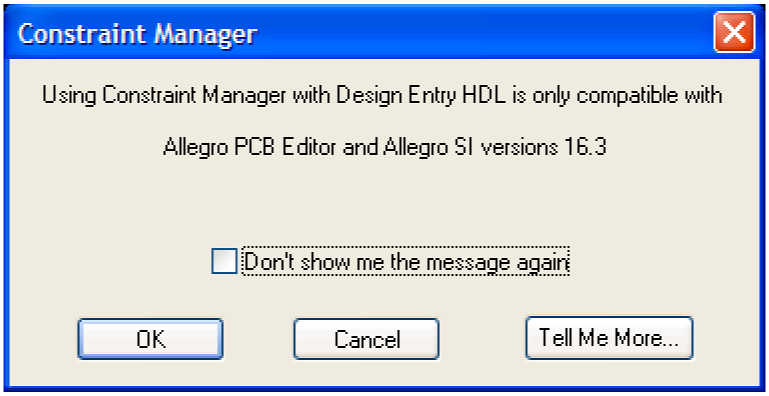
Yes	Saves the currently active schematic.
Yes All	Saves all the currently open schematics that have been modified.

No Does not save the currently active schematic.

No All Does not save any open schematic that has been modified.

Cancel Cancels the exit operation.

Compatibility With PCB Editor and Allegro SI



Using Constraint Manager with Design Entry HDL is only compatible with Allegro PCB Editor and Allegro SI versions 16.3

When you use Constraint Manager connected to Design Entry HDL with Allegro PCB Editor or Allegro SI, make sure that they belong to the same release version. For example, if you are using Constraint Manager connected to Design Entry HDL 16.3, you must use only Allegro PCB Editor or Allegro SI versions 16.3.

This is because Constraint Manager database is not backward compatible with earlier versions of PCB Editor or SI. For more information, see the *Allegro Design Entry HDL Constraint Manager Enabled Flow Guide*.

Design Entry HDL Options-Keys

Use these setup options to establish Design Entry HDL editor default settings for function keys.

Mode Specifies CTRL and SHIFT functions.

Fkey_number Specifies the default assignments for keys F1–F12.

Note: You cannot redefine F1 (help) or F10.

Customizing Design Entry HDL

You can customize toolbars, commands, menus, and keys in Design Entry HDL using *Tools – Customize*.

For more details, see:

- Customizing Toolbars
- Adding Buttons to Toolbars
- Customizing Commands
- Customizing Menus
- Customizing Keys

The customization changes you make for commands, menus, and keys binding are stored in ASCII files. The default files are available at <`your_install_dir`>/`share`/`cdssetup`/`concept`. The customizations are stored in $HOME/`cdssetup`/`concept`.

On the Windows platform, the HOME variable must be set in your local area. If you do not define the HOME variable, the customizations are not stored.

The toolbar customization information is stored in the registry.

Setting the HOME variable

On the Windows platform, the HOME variable must be set in your local area. If you do not define the HOME variable, the customizations are not stored.

The toolbar customization information is stored in the registry.

To set the HOME variable:

1. From the Start menu, choose *Settings – Control Panel.*

   The Control Panel appears.

2. Double-click the *System* icon.

   The System Properties dialog box appears.

3. Select the *Environment* tab.

4. In the User Variables section, enter HOME as a variable and your local work area (C:\ or D:\) as the value.

Design Entry HDL copies over `cdssetup/concept/concepthdl_cmd.txt`, `concepthdl_key.txt`, `concepthdl_menu.txt` to your HOME.

Customizing Toolbars

Related Info

Design Entry HDL allows you to change the position of all toolbars by dragging and dropping them. You can also design your own custom toolbars in Design Entry HDL.

1. Choose *Tools – Customize...*

   The *Customize* dialog box appears.

2. Select the *Toolbars* tab.

3. Click *New.*

   The *New Toolbar* dialog box appears.

4. Enter the name of the new toolbar you want to create.

5. Click *OK*.

   Design Entry HDL creates a new toolbar with the name you specified in the top left corner of the window.

   Note: If you create another new toolbar, Design Entry places the second toolbar over the first one.

6. Drag and drop buttons from any toolbar to the new toolbar. You can also select the *Buttons* tab to drag and drop buttons from here.

7. Click *OK* to apply and close the *Customize* dialog box.

Note: If want to reset an available toolbar to it's original setting, select the toolbar name in the Toolbars group and click Reset. If you select a custom toolbar you have created, the *Reset* button changes to *Delete*. To delete a custom toolbar, select the toolbar and click *Delete*.

Tips

- You can also customize custom toolbars by pressing CTRL+ALT and dragging and dropping any available button from any existing toolbar. You need not open *Tools – Customize* to do this.
- You can add and remove buttons from existing toolbars. To add a button, press ALT and drag and drop the desired button to the custom toolbar.
- To delete a button from a toolbar, press ALT and drag and drop the desired button into an empty area in Design Entry HDL. You can also add buttons to any available toolbar from the set of available buttons.
- To delete a button from a toolbar, press ALT and drag and drop the desired button into an empty area in Design Entry HDL.

Docking and Undocking Toolbars

To dock a toolbar on Windows

➤ Double-click the titlebar of the undocked toolbar.

To dock a toolbar on Solaris

➤ Click on the gripper in the toolbar and drag it to the desired location in the user interface.

If you have turned off the Cool Look check-box in the Customize Toolbars dialog box, the gripper will not be visible on the toolbars. In this case, click on the vacant space in the toolbar (Do not click on the title bar or a button), and drag it to the desired docking location in the user interface.

To undock a toolbar

- Click on the top window border in Design Entry HDL and move the docked toolbar out of Design Entry HDL.

  OR

- Click on the gripper on the left column of the toolbar, then drag and drop it for undocking.

  OR

- Click the right mouse button on the left column of the toolbar. A menu appears that contains two options - Allow Docking and Hide. If you unmark the Allow Docking option, the toolbar is undocked, and you cannot re-dock it. To dock the toolbar, you have to edit the registry.

If you turned off the Cool Look check-box, the gripper will not be visible. In this case, double-click on the vacant space in the toolbar.

Adding Buttons to Toolbars

Related Info

1. Choose *Tools – Customize.*

   The *Customize* dialog box appears.

2. Select the *Buttons* tab. This is selected by default.

3. Select the set of buttons you want to use from the Category list box.

   The Description box displays the Design Entry HDL function associated with the button.

4. Drag and drop the button to any area in Design Entry. If the button is dropped on an existing toolbar, Design Entry HDL adds the new button to it. If the button is dropped in an open area and not on a toolbar, Design Entry HDL creates a new toolbar with the new button.

Tips

- You can also customize custom toolbars by pressing CTRL+ALT and dragging and dropping any available button from any existing toolbar. You need not open *Tools – Customize* to do this.
- You can add and remove buttons from existing toolbars. To add a button, press ALT and drag and drop the desired button to the custom toolbar.
  To delete a button from a toolbar, press ALT and drag and drop the desired button into an empty area in Design Entry HDL.
- To delete a button from a toolbar, press ALT and drag and drop the desired button into an empty area in Design Entry HDL.

Customizing Commands

Related Info

Design Entry HDL allows you to add your own commands. These commands you define can be used to make custom menus and define custom keys. The custom commands you define cannot be used as console window commands.

To add a custom command

1. Choose *Tools – Customize...*

   The *Customize* dialog box appears.
2. Select the *Commands* tab.
3. Click *New* to enter a new command name in the *New Command* dialog box, and click *OK*.
4. Enter a *Short Description*.

   Design Entry HDL uses this text as the menu label when you insert the command as a menu. If you do not specify Short Description, Design Entry HDL takes the command name as the menu label.
5. Enter *Long Description*.

   Design Entry HDL displays this in the status bar when you move the cursor over the custom menu option.
6. Enter *Command String*.

   This is the console command that is to be bound to the new command.
7. Click *OK*.

Note: When you select any custom command in the Command Name list box, Design Entry HDL enables the Delete button.

The customizations you make on commands are stored in an ASCII file named `concepthdl_cmd.txt`. The default file is available at <*`your_install_dir`*>/`share`/`cdssetup/concept`. The customizations are stored in $HOME/`cdssetup/concept`.

Example of a command definition file

```
/* concepthdl_cmd.txt file, written 06/08/98 11:49:11 */
FileVersion 1;
WITH_SPACE “has space””has spaces””zoom fit”;
```

The syntax is as follows:

```
command-name short-description long-description command-string;
```

Setting the HOME variable

On the Windows platform, the HOME variable must be set in your local area. If you do not define the HOME variable, the customizations are not stored.

The toolbar customization information is stored in the registry.

To set the HOME variable

1. From the Start menu, choose *Settings – Control Panel*

   The Control Panel appears.

2. Double click on the System icon.

   The System Properties dialog box appears.

3. Select the Environment tab.

4. In the User Variables section, enter HOME as a variable and your local work area (C:\ or D:\) as the value.

   Design Entry HDL copies over cdssetup/concept/`concepthdl_cmd.txt` to your HOME.

Customizing Menus

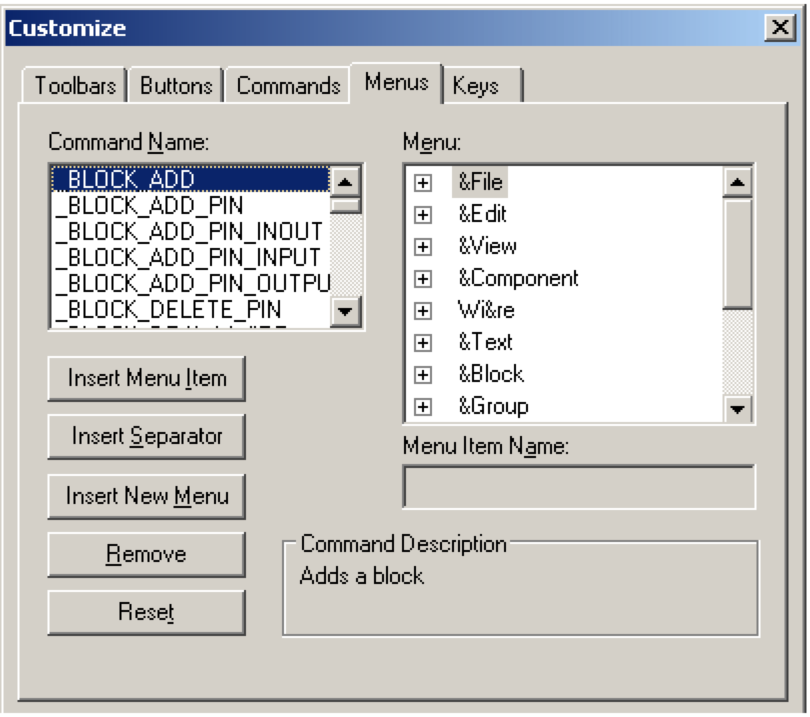
Related Info

Design Entry HDL allows you to customize your menus. You can

- Add a new menu with custom menu options under it.
- Add a menu option under an existing menu.

Customize Dialog box - Menus Page

You customize Design Entry HDL menus from the Menus tabbed page of the Customize dialog box:



Command Description
Adds a block

Command Name — Displays a list of Design Entry HDL commands. You can customize the Design Entry HDL menu to include corresponding menu items for each of these commands. For example, you can add a corresponding menu item for the _GROUP_SMASH command, which breaks components into their individual elements.

Menu — Displays a list of menus. This original menu list is picked from the default `concepthdl_menu.txt` file available at <*your_install_dir*>/`share/cdssetup/concept`. You can customize the Menu list to add or delete menu items to the original menu list.

Command Description — Shows a description of the command selected n the Command Name list.

Insert Menu Item

Inserts a corresponding menu item for the commands in the Command Names list.

Insert Separator

Creates a line separator between the menu selected in the Menu pane and the next menu.

Insert New Menu

Inserts a new menu item under the currently selected menu item in the Menu list.

Remove

Removes the currently selected menu item in the menu list.

Reset

Resets the menus to their original settings (factory settings) stored in the default `concepthdl_menu.txt` file at. `<your_install_dir>/share/cdssetup/concept.` All the menu customizations stored in the `concepthdl_menu.txt` file stored in your HOME/cdssetup/concept directory are lost when you reset menus.

Note: If you click this button when the Windows mode is enabled, the menus will be reset to the Windows mode menus, which are different from the menus in the non-Windows mode. The original settings for menus in the Windows mode are stored in the `concepthdl_menu_win.txt` file at. `<your_install_dir>/share/cdssetup/concept.`

To insert a custom menu

1. Choose *Tools – Customize.*

   The *Customize* dialog box appears.

2. Select the *Menus* tab.

3. Select the command you want to add from the Command Name list box.

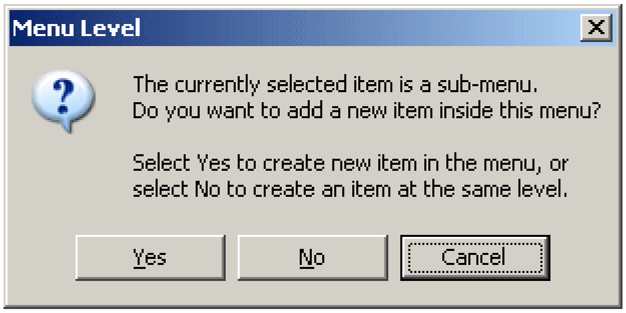
   All the Design Entry HDL standard commands are displayed with leading underscore character.

4. Select the menu under which you want to add the command from the *Menu* list box.

   In this tree control view, select the + and – marks to expand and collapse a level in the menu hierarchy.

5. Click the *Insert Menu* button.

   Design Entry HDL asks you whether you want to insert the menu at the primary level (menu bar in Design Entry HDL), or within a menu.



Menu Level

The currently selected item is a sub-menu.
Do you want to add a new item inside this menu?

Select Yes to create new item in the menu, or
select No to create an item at the same level.

6. Click *Yes* if you want to insert the command within the selected menu or click *No* to Insert the command as a separate menu in the same level as that of the selected menu.
7. Click *OK*.

Note: Design Entry HDL makes appropriate changes to the *concepthdl_menu.txt* file. Design Entry HDL reads this file every time you invoke it. The customizations you make for menus are stored in an ASCII file named `concepthdl_menu.txt` in the $HOME/cd`ssetup/ concept/` directory.

Excerpt from a menu definition file

```
/* concepthdl_menu.txt file, written 04/30/99 18:02:21 */
FileVersion 1;
Window Schematic {
    "&File" {
        "&New" _FILE_NEW;
        "&Open..." _FILE_OPEN;
        "&Close" _FILE_CLOSE;
        SEPARATOR;
        "&Save" _FILE_SAVE;
        "Save &As..." _FILE_SAVEAS;
        SEPARATOR;
        "&Revert" _FILE_REVERT;
        "Reco&ver..." _FILE_RECOVER;
        "Re&move..." _FILE_REMOVE;
        SEPARATOR;
        "Edit Pa&ge/Symbol" {
            "&Next" _FILE_NEXT_PAGE;
            "&Previous" _FILE_PREV_PAGE;
            "&Go To..." _FILE_GOTO_PAGE;
```

The syntax of the definition portion of the file is:

```
Window Schematic{ menu-statements}
     menu statements =: sub-menu-definitions,
     menu-command-definitions, or separator-definitions
     sub-menu-definitions =: menu-text {menu-statements}
     men-command-definitions =: menu_text command-name;
     sepaator-definitions =:SEPARATOR;
```

Double quotes usually surround menu-text.

Setting the HOME variable

On the Windows platform, the HOME variable must be set in your local area. If you do not define the HOME variable, the customizations are not stored.

To set the HOME variable

1. From the Start menu, choose *Settings – Control Panel.*

   The Control Panel appears.

2. Double click on the System icon.

   The System Properties dialog box appears.

3. Select the Environment tab.

4. In the User Variables section, enter HOME as a variable and your local work area (C:\ or D:\) as the value.

   Design Entry HDL copies over `cdssetup/concept/concepthdl_menu.txt` to your HOME.

Customizing Keys

Related Info

Design Entry HDL allows you to customize shortcut keys. You can add custom keys and bind them to the Design Entry HDL standard commands or to your own commands.

To define a new key, do the following:

1. Choose *Tools – Customize.*

   The *Customize* dialog box appears.

2. Click the *Keys* tab.
3. Select a command from the *Command Name* list.
4. Place cursor in the *Press New Key* field and press the shortcut keystroke you want to define.
5. Click *Add Key*.

   The shortcut key is added to the *Keys* list.

6. Click *OK*.

Note: Clicking the *Reset* button restores all the shortcut keys to default. You can remove a particular shortcut key by selecting the key combination in the *Keys* list and clicking the *Remove* button.

Key combinations	**Supported on**
CTRL+<number>	Windows
CTRL+<letter>	Windows
CTRL+<function key> except F1	Windows
CTRL+ <arrow key>	Windows
CTRL+ Insert/Home/Delete/End	Windows
SHIFT+<function key> except F1	Windows
SHIFT+ Insert/Home/Delete/End	Windows
SHIFT+ <arrow key>	Windows
ALT+<number>	Windows
ALT+<letter>	Windows
ALT+<function key> except F1	Windows
ALT+ <arrow key>	Windows
ALT+ Insert/Home/Delete/End	Windows
Insert/Home/Delete/End	Windows

Arrow keys	Windows
Function Keys except F1	Windows

The customized changes you make to a key binding are stored in an ASCII file named `concepthdl_key.txt`. The default file is available at <*your_install_dir*>/`share`/`cdssetup`/`concept`. The customizations are stored in $HOME/`cdssetup`/`concept`.

Excerpt from a key definition file:

```
/* concepthdl_key.txt file, written 06/10/16 12:05:44 */
FileVersion 1;
Ctrl+C _EDITTEXT_COPY;
Ctrl+F _TOOLS_GLOBAL_FIND;
```

The syntax of a statement is

```
Key-definition command-name;
```

To set the HOME variable:

On the Windows platform, the HOME variable must be set in your local area. If you do not define the HOME variable, the customizations are not stored.

To set the HOME variable, do the following:

1. Access the Environment Variables dialog using the navigation specific to your Windows operation system.
2. In the User Variables section, enter HOME as a variable and your local work area (C:\ or D:\) as the value.

   Design Entry HDL copies cdssetup/concept/`concepthdl_key.txt` to your HOME.

Assign Power Pins

Procedure

Use this dialog box to view and assign properties to power pins of a component.

Pin No.	Lists the power pin numbers of the component.
Power Pins	Lists the names of power pins defined in the `chips.prt` file of the component.
Power Names	Lists the names of power pins defined through the `POWER_PINS`, `MERGE_POWER_PINS`, and `POWER_GROUP` properties on the instance or symbol of the component. You can change the name of the power pin by selecting a global signal from the drop-down list or giving a name of your own choice. The global signals present at lower levels of hierarchy appear in the drop-down list only after the design is packaged.
NC Pins	Lists the NC pins of the component defined through the `NC_PINS` and `MERGE_NC_PINS` properties in the `chips.prt` file or the instance or the symbol of the component.
Separate property for each pin name	Select this check box to create a separate `POWER_PINS` property for each pin name.
Specify maximum property length	Select this check box to specify the maximum number of characters in the properties created by Design Entry HDL. Design Entry HDL then splits the property value and creates more than one properties.
OK	Propagates the changes that you have made in the dialog box to the instance of the component.
Cancel	Does not propagate the changes made in the dialog box to existing properties of the component.

Group Controls

Use this dialog box to perform various operations on groups of objects.

Group

Type the name of the group or scroll to select the group name. The group name can be from A to Z. This group becomes the currently active group.

Contents

Click to open the Group Contents dialog box.

The Group Contents dialog box displays the objects in the currently active group.

Operation

Select from *Create*, *Include*, *Exclude*, and *Flush*.

Create

Use *Create* to form a new group with the name selected in the *Group* field constituting objects of types selected in the *Object types* field.

Include

Use *Include* to add an object or a group to the currently active group.

Exclude

Use *Exclude* to remove the following from the currently active group:

- objects of the specified *Object types*
- objects in other group(s).

Flush

Use *Flush* to remove objects of the specified *Object types* from the currently active group*.*

Mode

The values of the *Mode* field depend on the *Operation* being performed.

- Create operation

  Select *Mode* as *Rectangles*, *Polygons*, *Select All*, or *Expression*.

  Rectangles: forms a group of objects within a stretchable rectangle. When you have captured objects in a rectangle, click *Done* to form the group.

  Polygons: forms a group of objects within a polygon. When you have captured objects in a polygon, click *Done* to form the group.

  Select All: includes all objects on the current page.

  Expression: matches a pattern string with objects on the current page and groups them together.

- Include operation

  Select *Mode* as *Points* or *Group Names*.

  Points: Includes the objects of specified *Object types* into the current group. When you have selected the objects on the canvas, click *Done* to include them in the group.

  Group Names: Specify group name(s). The objects of specified *Object types* in these group(s) are added into the current group.

- Exclude operation

  Select *Mode* as *Points* or *Group Names*.

  Points: Excludes the objects of specified *Object types* from the current group. When you have selected the objects on the canvas, click *Done* to exclude them from the group.

  Group Names: Specify group name(s). The objects of specified *Object types* in these group(s) are excluded from the current group.

- Flush operation

  The objects of the specified *Object types* are removed from the currently active group.

Object Types	Determines the types of objects on the current page to consider while performing a group operation on them. Select from *All types*, *Bodies*, *Wires*, *Conns*, *Props*, and *Nets*.
Execute/ Done	Performs the specified operation on the objects on the current page.
Close	Dismisses the Group Controls dialog box.

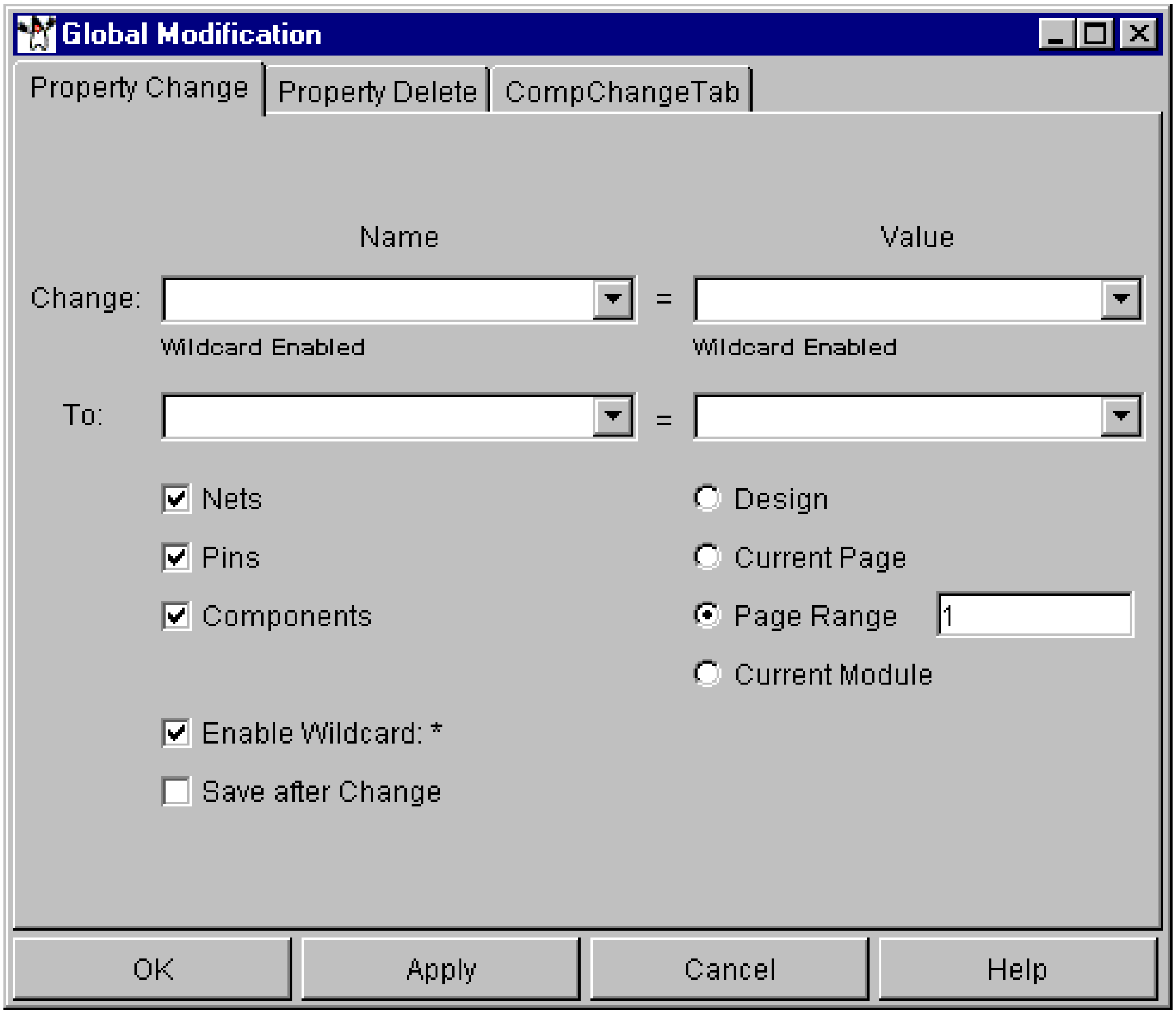
Text Set Size

Use this dialog box to specify the text size to be used for displaying the properties and notes in the currently active group.

Global Modification – Property Change

Procedure

Use these options to modify component, pin, and net properties across a design.



Change: =

To: =

☑ Nets

☑ Pins

☑ Components

○ Design

○ Current Page

● Page Range 1

○ Current Module

☑ Enable Wildcard: *

☐ Save after Change

Field	Description
Change (Name)	Specifies the name of the property to be changed. You can either type in the property name or select a previously entered name from the drop-down list. It is a case-insensitive field. After a new property is processed, the name is added to the top of the selection list, and is available for the current Design Entry HDL session. When you close the Editor, any additions to the property name selection list are lost.

Wildcard support

Wildcards are supported in the GUI for the original property name and value. Wildcards are handled in the following ways:

- The `*` is always handled as a wildcard in the original property name.
- The *Enable Wildcard* check box controls whether the `*` in the original property value is handled as a wildcard or as a literal.

Field **Description**

To (Name)

Specifies the new name for the selected property in the *Change (Name)* field. You can either type in the new name or select a previously entered name from the drop-down list. It is a case-insensitive field.

After a new property is processed, the name is added to the top of the selection list, and is available only for the current Design Entry HDL session.

Wildcard support

Wildcards are not supported in the GUI for the new property name and value. Wildcards are handled in the following ways:

- The `*` is never allowed in the new property name.
- The `*` is always processed as a literal in the new property value.
- The following pull-down entry is available as the first element of the new property name combo box. It helps you to retain the original property name.

```
++Preserve Source Name++
```

- The following pull-down entry is available as the first element of the new property value combo box. It allows you to retain the original property value.

```
++Preserve Source Value++
```

- Do not select the `Preserve` options for the name and value in the same run. These options are available in the *Batch* mode by using the `<<PRESERVE>>` keyword instead of a property name or a value.

Note: To keep the property name unchanged, the value in this field must match the property name in the field above it.

Change (Value)

Specifies the value of the property on the schematic to be changed. The *Change (Value)* combo box provides a history of all values used from which you can select a value or type a new value.

Field	**Description**
To (Value)	Specifies the new property value. The *To (Value)* combo box provides a history of all values used from which you can select a value or type a new value. The `++Preserve Source Value++` option in the *To (Value)* drop-down list is used to retain the value from the source property. This is because the `*` character is always treated as a literal in the new property value field. **Note:** To keep the property value unchanged, the value in this field must match the property value in the field above it.
Enable Wildcard	Select this check box to control the usage of `*` in the original property value. If the check box is selected, the `*` in the original property value is handled as a wildcard. Otherwise, it is handled as a literal. By default, the *Enable Wildcard* check box is selected.
Nets	Select this check box to specify that a property on a net is being modified. When this check box is selected, properties that match the input criteria described above are modified provided they are attached to this type of Design Entry HDL object. If the check box is not selected, no modification is made even if the property name and value match the input criteria.
Pins	Select this check box to specify that a property on pin is being modified. When this check box is selected, properties that match the input criteria described above are modified provided they are attached to this type of Design Entry HDL object. If the check box is not selected, no modification is made even if the property name and value match the input criteria.
Components	Select this check box to specify that a property on a component is being modified. When this check box is selected, properties that match the input criteria described above are modified provided they are attached to this type of Design Entry HDL object. If the check box is not selected, no modification is made even if the property name and value match the input criteria.
Design	Controls the scope of modifications that are made. Select the *Design* radio button to iterate over all pages in all modules of the design.
Current Page	Limits the scope of modifications to the current page.

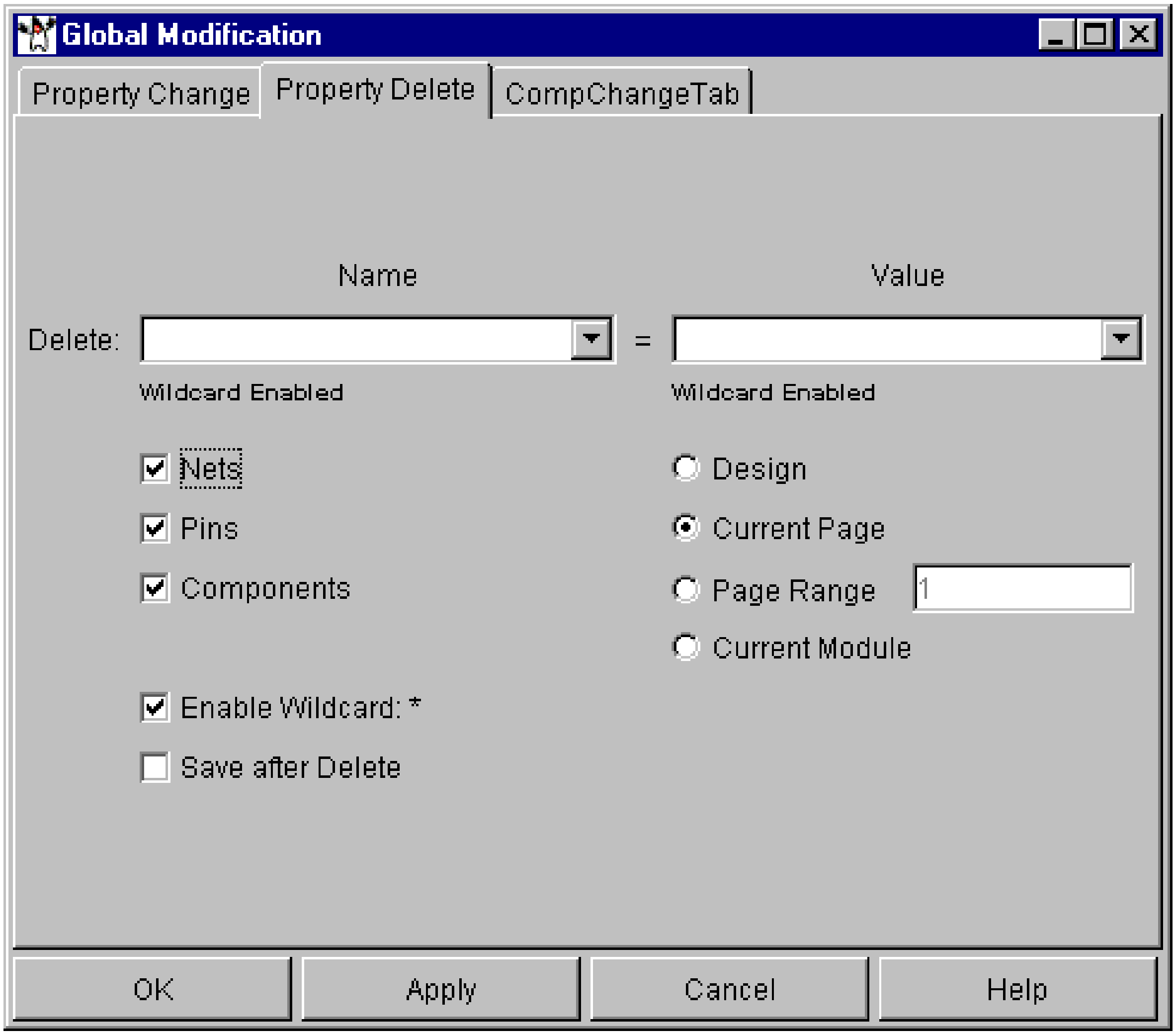
Field	**Description**
Page Range	Controls the scope of modifications that are made. Select the *Page Range* radio button to specify a comma-separated list of pages or page ranges. For example `1,3,5,7-12`.
Current Module	Controls the scope of modifications that are made. Select the *Current Module* radio button to process only the current module, instead of the current page or hierarchy.
Save after Change	Select this check box to specify that the schematic sheet be saved after a modification is made. Design Entry HDL allows only 50 drawing sheets to be open at one time. Therefore, large designs are not updated completely unless the *Save after Change* option is selected.
OK	Initiates the processing and closes the Global Modification window. After all the changes are made, a *Summary* screen appears. The *Summary* screen lists the current status of the design, which includes the number of pages processed, the number of properties changed, and so on.
Apply	Saves all the changes you made without closing the Global Modification window.
Cancel	Closes the Global Modification window and makes no changes to the schematic.

Note: Design Entry HDL supports default properties added to library symbols. The Global Modification window cannot delete or make changes to these default symbol properties.

Global Modification – Procedure Delete

Procedure

Use these options to delete component, pin, and net properties across a design.



☑ Nets
☑ Pins
☑ Components
○ Design
● Current Page
○ Page Range 1
○ Current Module
☑ Enable Wildcard: *
□ Save after Delete

Field	**Description**
Delete (Name)	Specifies the name of the property to delete. You can either type in the property name or select a previously entered name from the drop-down list. It is a case-insensitive field. After a new property is processed, the name is added to the top of the selection list, and is available for the current Design Entry HDL session. When you close the Editor, any additions to the property name selection list are lost. ***Wildcard support*** Wildcards are supported in the GUI for the delete property name and value. The `*` is always handled as a wildcard in the original property name.
Delete (Value)	Specifies the value of the property on the schematic to be deleted. The *Delete (Value)* combo box provides a history of all values used, from which you can select a value, or type a new value. ***Wildcard support*** The *Enable Wildcard* check box controls whether the `*` in the delete property value is handled as a wildcard or as a literal.
Enable Wildcard	Controls how the `*` in the delete property value is handled. If the check box is selected, the `*` in the delete property value is handled as a wildcard. Else, the `*` in the delete property value is handled as a literal. By default, the *Enable Wildcard* check box is selected.
Nets	Select this check box to specify that a property on a net is being deleted. When this check box is selected, properties that match the input criteria described above are deleted provided they are attached to this type of Design Entry HDL object. If the check box is not selected, no modification is made even if the property name and value match the input criteria.

Field	**Description**
Pins	Select this check box to specify that a property on a pin is being deleted. When this check box is selected, properties that match the input criteria described above are deleted provided they are attached to this type of Design Entry HDL object. If the check box is not selected, no modification is made even if the property name and value match the input criteria.
Components	Select this check box to specify that a property on a component is being deleted. When this check box is selected, properties that match the input criteria described above are deleted provided they are attached to this type of Design Entry HDL object. If the check box is not selected, no modification is made even if the property name and value match the input criteria.
Design	Controls the scope of modifications that are made. Select the *Design* radio button to iterate over all pages in all modules of the design.
Current Page	Limits the scope of modifications to the current page.
Page Range	Controls the scope of modifications that are made. Select the *Page Range* radio button to specify a comma-separated list of pages or page ranges. For example `1,3,5,7-12`.
Current Module	Controls the scope of modifications that are made. Select the *Current Module* radio button to process only the current module, instead of the current page or hierarchy.
Save after Delete	Select this check box to specify that the schematic sheet be saved after a modification is made.
OK	Initiates the processing and closes the Global Modification window.
Apply	Saves all the changes you have made without closing the Global Modification window.
Cancel	Closes the Global Modification window and makes no changes to the schematic.

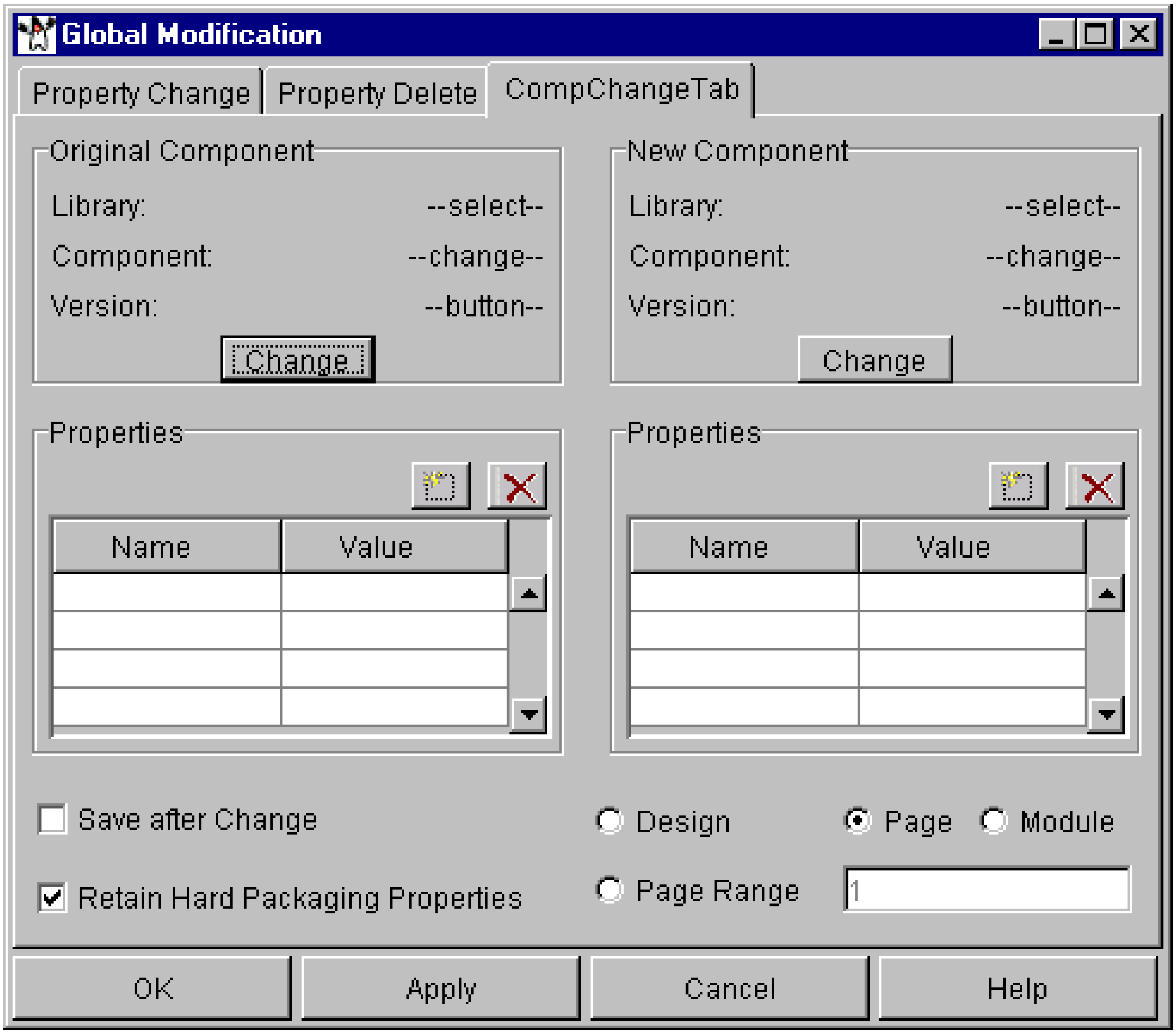
Global Modification – Component Change

Procedures

Procedures

- Selecting the component to replace from the physical part filter
- Selecting the new component from the physical part filter
- Selecting the component to replace from the design (Schematic Pick)
- Selecting the new component from the design (Schematic Pick)

Use these options to delete, modify, replace, and refresh components globally.



☐ Save after Change

☑ Retain Hard Packaging Properties

○ Design ● Page ○ Module

○ Page Range 1

Group	Description
Original Component	Use the fields in this group to select the original component either from the physical part filter or the design.
Library	Represents the library the component belongs to.
Component	Represents the name of the component.
Version	Represents the version of the component.

Group	Description
New Component	Use the fields in this group to select the new component from from the physical part filter or the design.
Library	Represents the library the component belongs to.
Component	Represents the name of the component.
Version	Represents the version of the component.
Properties	Lists the properties of the component selected from the design. For example, the *Properties* list below the *Original Component* group lists the properties of the selected original component. Similarly, the *Properties* list below the *New Component* group lists the properties of the selected new component. Each property in the *Properties* list is represented by a Name-Value pair. You can also add a new property to the list and delete a selected property from the list using the *New* and *Delete* icons in the *Properties* list.
Processing Options	Use the fields in this group to define the processing options.
Retain Hard Packaging Properties	Select this check box to specify that hard packaging properties, if any, are retained during processing. Hard packaging properties, such as LOCATION and PN, are properties entered by the user. Soft packaging properties are generated by the tool itself and are prefixed with a $sign, for example $LOCATION and $PN. This option only pertains to the *global replace* feature. If pin locations of the source and destination components are the same, you have the option to retain the hard packaging data. If the pin locations are different, the location is retained, however, pin numbers and section information are not retained.
Save after Change	Select this check box to specify that the schematic sheet be saved after a component is changed.
Design	Controls the scope of modifications that are made. Select the *Design* radio button to iterate over all pages in all modules of the design.
Current Page	Limits the scope of modifications to the current page.
Page Range	Controls the scope of modifications that are made. Select the *Page Range* radio button to specify a comma-separated list of pages or page ranges. For example `1,3,5,7-12`.

Group	Description
Current Module	Controls the scope of modifications that are made. Select the *Current Module* radio button to process only the current module, instead of the current page or hierarchy.

You can also clear a component selection by right-clicking the table in the *Properties* group box. There are two options in the shortcut menu, *Clear Table* and *Clear Table & Component*. Selecting the *Clear Table* option clears the table contents and selecting the *Clear Table & Component* option clears the table and Lib/Cell/View contents.

Model Assignment

Use this dialog box to assign signal models to components in a design. The Model Assignment window provides a convenient way to assign models for IC devices and auto generate these models for discrete components. You can assign signal models to components and pins using the new Model Assignment window. You can assign signal models to multiple components, simultaneously.

The Model Assignment window is a three-pane window. The first pane lists the names of the parts used in the design, the second pane lists details about the part selected in the first pane, and the third pane, which is hidden by default, lists the pin details of the part instance selected in the second pane.

See Assigning Signal Models section of the *Allegro Design Entry HDL-Constraint Manager User Guide* for more information.

Field	Description
First Pane	Shows a list of all the components used in the design.
Name	Shows the physical part names of all the electrical components used in the design. The first component is selected and a detailed list of instances of the selected components is listed on the second pane.
Total	Shows the number of instances of each component.
Errors	Shows the total number of instances which have incorrect signal models assigned to them.

Unassigned
Shows the number of instance that do not have a signal model assigned to them.

Second Pane
Shows a detailed list of information about all the instances of the selected component in a grid format.

Instance
Shows the name of the part instance.

Block
Shows the block name of the part instance.

RefDes
Shows the location of the part instance.

SI Model
Shows the signal model assigned to the component instance.

Part
Shows key properties of the selected component instance from the physical part table (ppt). Shows (No Info) for a part instance without an entry in the ppt.

Model Src
Shows the source of the signal model, whether schematic or opf.

Third Pane
Allows you to add pin buffer models to the pins. It displays detailed information about the pins of the component instance selected in the second pane. This pane is hidden by default. To display this pane, you need to click the *IO Pin Details* button on the window.

Name
Shows the pin name, pin type, and signal model (if assigned).

SI Model
Shows the signal model assigned to the pin.

PN
Shows the PN property representing the physical pin number of the selected pin of the component instance.

Type
Shows the pin type of the selected pin.

Auto Generate
Click this button to automatically generate signal models for discrete components in a design (resistor, inductor, or capacitor). This button is activated when you select a discrete component in the component list on the first pane.

Setup
Click this button to setup the path to the device model library (dml) file before assigning signal models. The `.dml` file contains information about the models that you can assign to components and pins on a schematic.

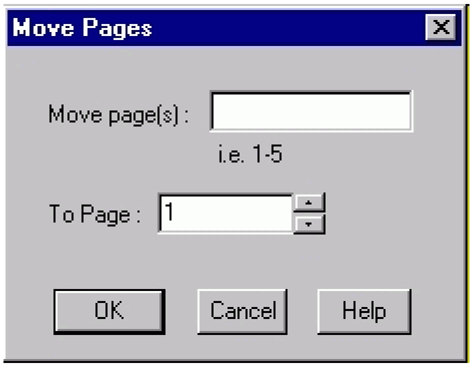
IO Pin Details
Click this button to display detailed information about the pins on a specific component.

Refresh	Click this button to reload the Model Assignment window with the updated information of the components in the design. Helps in synchronizing the Model Assignment window with Design Entry HDL, in case you make any changes in Design Entry HDL.
Apply	Click this button to apply the changes done in the Model Assignment window to the schematic. Use this button to update the schematic.
Status Bar	Displays the status of the model assignment validation. If the validation routine fails, a message is displayed in the status bar stating that the signal model does not map to the component instance.
Details	Click this button to display the Error Details message box, showing a detailed list of the errors generated by the validation routine performed after signal model assignment.

Move Pages

Procedure   Command

Use this dialog box to move a page or a set of pages within a schematic. You can move pages between existing pages of a schematic. You can also move a set of pages to a non-existent location and move non-contiguous pages to contiguous locations.



Field **Description**

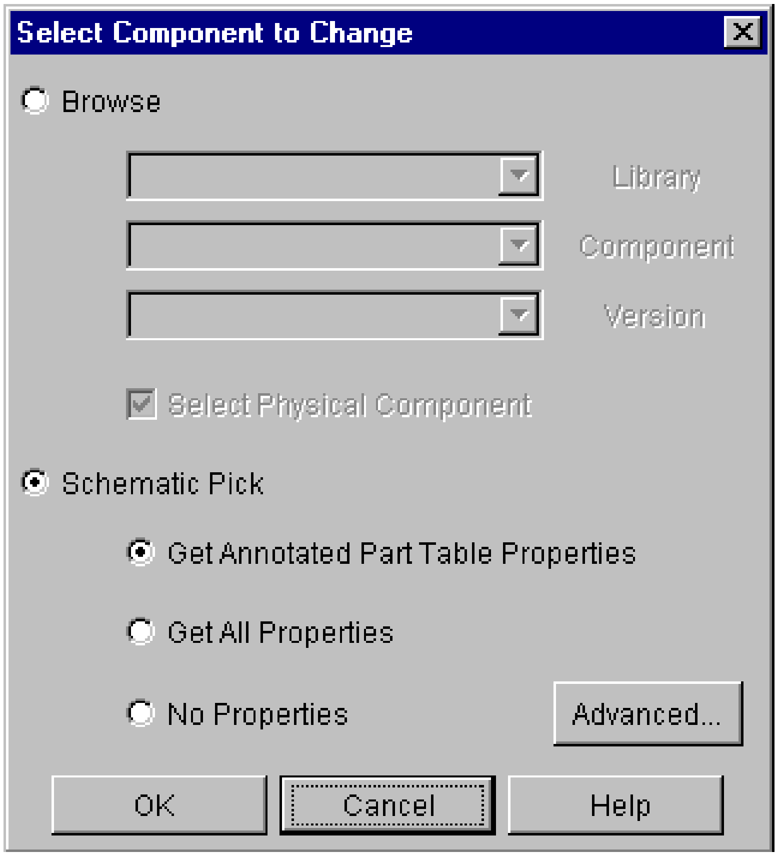
Move page(s)	Indicates the page or the set of pages that you want to move.
To Page	Is the location in the schematic where the pages will be moved. For example, if you specify 5 the moved pages will precede the current page 5.

For more information, refer to Performing Page Management Operations.

Select Component to Change

You can select a component to change by either browsing the libraries using the combo boxes or by selecting the component from the schematic.

When you click the *Change* button in the *Original Component* group of the Global Modification window, the *Select Component to Change* dialog box appears. This dialog box is used to select the component you want to replace.

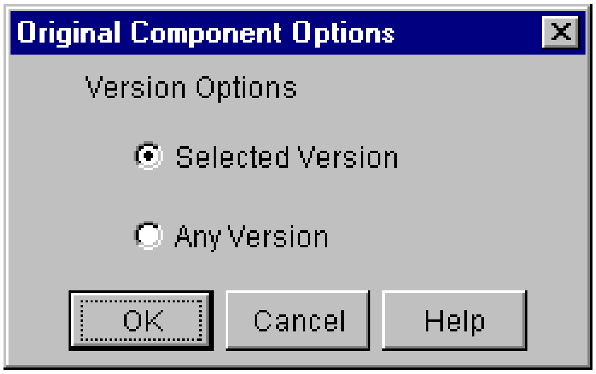


Field	Description
Browse	
Library	Use the *Library* drop-down arrow to select the library of the component you want to replace.
Component	Use the *Component* drop-down arrow to select the component.
Version	Use the *Version* drop-down arrow to select the version of the component.

Field	Description
Select Physical Component	Select this check box to specify that the component search is qualified with a set of physical properties. This ensures that only a specific physical component on the design is changed. **Note:** If this check box is not selected, changes are made only to the logical component.
Schematic Pick	
Get Annotated Part Table Properties	Select this radio button to retrieve only the annotated part table properties of the original component. When this option is selected, Part Information Manager is launched with the selected component highlighted. Select *OK* in Part Information Manager so that the *Global Modification* UI can show the properties that are key in the part table and the properties that are user properties. **Note:** Sometimes, libraries change and components are not found when Design Entry HDL launches and reads a schematic. A solution might be to replace these components with a new library component. In this situation, the *Get Annotated Part Table Properties* option does not work because the component is not physically available in the library.
Get All Properties	Select this radio button to retrieve all properties of the original component from the schematic. This is helpful when you have user properties on a component and want to retain the properties for component qualification.
No Properties	Select this radio button so that no properties of the original component are retrieved.

Original Component Options

When you click the *Advanced* button in the Select Component to Change dialog box, the Original Component Options dialog box appears.



Version Options

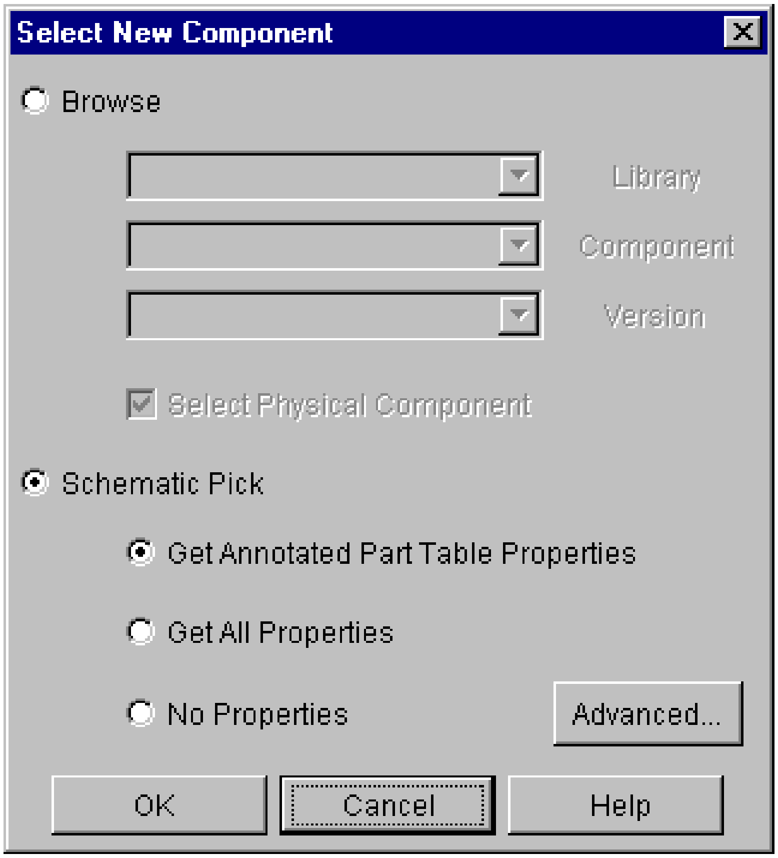
● Selected Version

○ Any Version

Field	Description
Version Options	
Selected Version	Select this radio button to replace instances of only the selected version of the original component.
Any Version	Select this radio button to replace instances of all versions of the original component. When using wildcards, you can create cases where you can use replace, modify, and refresh in conjunction. For instance, if you change `RES.* to RES.1`, and If there are three versions of the `RES` symbol, each with a different pinout, the following will occur: RES version1 change to RES version1 MODIFY RES version2 change to RES version1 MODIFY RES version3 change to RES version1 MODIFY

Select a New Component

When you click the *Change* button in the *New Component* group of the Global Modification window, the *Select New Component* dialog box appears. This dialog box is used to select the new component.

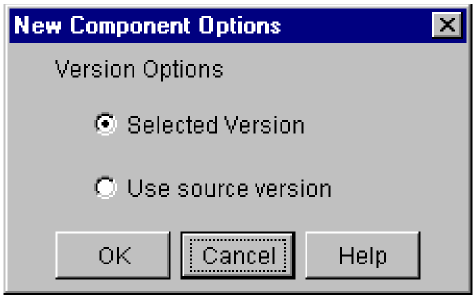


Field	Description
Browse	
Library	Use the *Library* drop-down arrow to select the library of the component you want to replace.
Component	Use the *Component* drop-down arrow to select the component.
Version	Use the *Version* drop-down arrow to select the version of the component.

Field	Description
Select Physical Component	Select this check box to specify that the changes are made to the physical component. **Note:** If this check box is not selected, changes are made only to the logical component.
Schematic Pick	
Get Annotated Part Table Properties	Select this radio button to retrieve only the annotated part table properties of the new component.
Get All Properties	Select this radio button to retrieve all properties of the new component from the schematic.
No Properties	Select this radio button to retrieve no properties of the new component.

New Component Options

When you click the *Advanced* button in the Select New Component dialog box, the *New Component Options* dialog box appears.



New Component Options

Version Options

● Selected Version

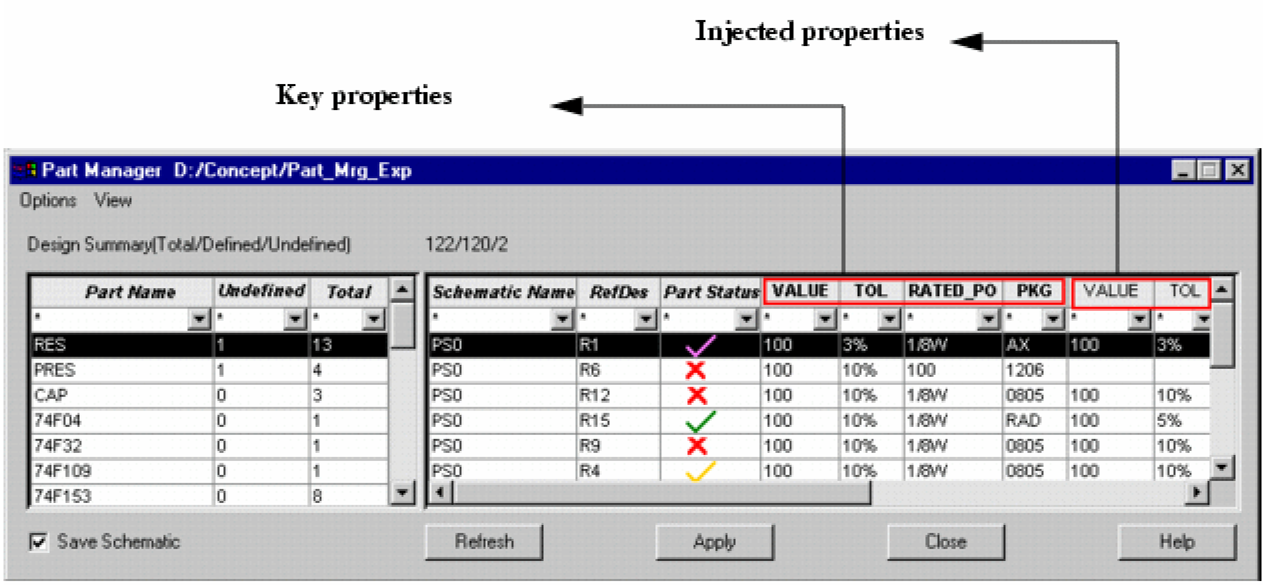
○ Use source version

Field	Description
Version Options	
Selected Version	Select this radio button to replace the original component with the selected version of the new component.
Use source version	Select this radio button to replace the original component with the source version of the new component.

Part Manager

When you run the partmgr command, the Part Manager window appears. The Part Manager window consists of two panes:

- Left pane - This pane lists the names of the parts used in the design, total number of instances of the part, and the instances that are out of sync with the corresponding ptf. The left pane also lists the summary of part instances indicating the number of defined and undefined part instances.
- Right pane - This pane displays a detailed grid of part information including the key and injected properties, and the part status.



Injected properties

Key properties

Part Manager D:/Concept/Part_Mrg_Exp

Design Summary(Total/Defined/Undefined) 122/120/2

Part Name	Undefined	Total
*	*	*
RES	1	13
PRES	1	4
CAP	0	3
74F04	0	1
74F32	0	1
74F109	0	1
74F153	0	8

Schematic Name	RefDes	Part Status	VALUE	TOL	RATED_PO	PKG	VALUE	TOL
*	*	*	*	*	*	*	*	*
PS0	R1	✓	100	3%	1/8W	AX	100	3%
PS0	R6	✗	100	10%	100	1206		
PS0	R12	✗	100	10%	1/8W	0805	100	10%
PS0	R15	✓	100	10%	1/8W	RAD	100	5%
PS0	R9	✗	100	10%	1/8W	0805	100	10%
PS0	R4	✓	100	10%	1/8W	0805	100	10%

☑ Save Schematic

The following table lists the various fields and options of the Part Manager window with descriptions.

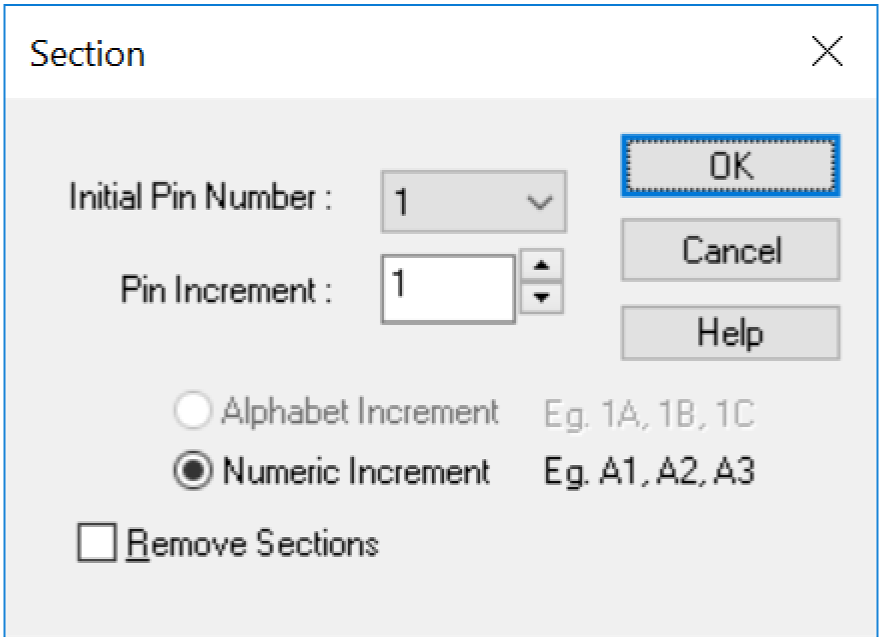
Fields	Description
Part Name	Lists the physical part name.
Undefined	Lists the number of instances which are out of sync with the corresponding ptf.
Total	Lists the total number of instances of a part used in a design.
Schematic Name	Shows the block name of the part instance.
RefDes	Shows the location of the part instance. If the location property is not available, a question mark ("?") is displayed in this column.
Part Status	Shows the status of parts represented by icons. The values that this column can take are: “MATCHED” (Green), “INJECTED MISMATCH” (Yellow), “NOT MATCHED” (Red), or “IGNORED”.
Filters	Displays specific part instances by applying filters. Each of the columns of the Part Manager window has filters that support wildcard characters. Each filter has a drop-down combo box that shows the valid values which you can use to filter out the required values.
Summary	Displays a summary of all the parts of the design and their status. The summary lists total number of parts, defined parts (“MATCHED” and “INJECTED MISMATCH”), and undefined (“NOT MATCHED”)parts. When you update an undefined part instance with a valid ptf entry and apply the changes, the summary is adjusted to reflect the current status of the design.
Save Schematic	Updates the schematic with the changes that you make in Part Manager. If this check box is not selected, changes are passed to schematic, but not saved. If you save the schematic, you will see the updated values.
Apply	Applies the changes to the schematic using Part Manager. Use this button to update the schematic.

Fields	Description
Refresh	Reloads Part Manager with the updated details of the parts used in the design. Helps in synchronizing Part Manager with Design Entry, in case you make any changes in Design Entry.

For more information, see Part Manager.

Section

Use this dialog box to assign pin numbers to multiple logical part instances. This dialog box helps you assign pin numbers depending on the increment you specify. The pin numbers you specify can be numeric or alphanumeric.



Section

Initial Pin Number : 1

Pin Increment : 1

○ Alphabet Increment Eg. 1A, 1B, 1C

● Numeric Increment Eg. A1, A2, A3

□ Remove Sections

Fields	Description
Initial Pin Number	Identifies the first pin number to be assigned to the first part instance. **Note:** The pin numbers displayed support multiple alphanumeric pin sorting (for example, A1-B1, A2-B2).

Fields	Description
Pin Increment	Identifies a number by which you want to increment the subsequent pin numbers.
Alphabet Increment	Increments pin numbers alphabetically. For example, if you select this check box and specify the increment as 1, pin numbers will be assigned as 1A, 1B, 1C, and so on.
Numeric Increment	Increments pin numbers numerically. For example, if you select this check box and specify the increment as 1, pin numbers will be assigned as A1, A2, A3, and so on.
Remove Selections	Enables you to unsection multiple part instances, simultaneously.

Bias Point Preferences

Fields	Description
Displayed Precision	Use this text box to specify the number of digits that are to be used for displaying bias point information. The value specified by default is 4. You can specify values from 2 to 5. For example, consider that the bias point voltage at a node is `7.6931` volts. If you specify 3 in the Displayed Precision text box, the value displayed in Design Entry HDL will be `7.69` volts. And if the value specified in the Displayed precision text box is 2, the value displayed on the schematic will be `7.7` volts.
Color	Use this drop-down list box to specify the color of the text used for displaying bias information on the schematic. For example, if you want bias voltages to be displayed in orange color, you should select orange from the Color drop-down list corresponding to the Voltage label.
Text Size	Use this spin box to specify the size of text to be used for displaying bias voltage, bias current, and bias power. Example: To change the size of the text used for displaying bias voltage, use the Text Size spin box next to the Voltage label.

Update Bias Point Information Automatically	Select this check box to automatically load bias point information on to Design Entry HDL after a design is simulated using PSpice.

Import Design

Procedure

Use this dialog box to specify the `project.cpm` file of the project or the `cds.lib` file associated with the project from which you want to import a sheet. You can import one or more schematic sheets to the currently open project from another project.

Project File (`.cpm`)/ Library Paths file (`.lib`)

Specify the path to the appropriate project file (`.cpm`) or the cds.lib file. You can also browse to the required file.

The Project list box displays the preferred list of projects saved in the `preferred_projects.txt` file and the projects accessed in the current session. You can quickly select an available project from the list box.

Paste Special

Procedure

Use this dialog box to specify whether you want to:

- Paste copied schematic parts on to the target schematic directly
- Change the signal names of the schematic before pasting them on the target location

Change Signal Names

Displays the Paste Special dialog box where you can rename the signals in the dialog box.

The old signal names are displayed in the O*ld Signal Names* column. You can specify a new signal name for an existing signal by entering the changed value in the corresponding cell for that signal in the New *Signal Names* column.

Paste Schematic

Pastes the copied part on to the target location in the schematic.

Add Prefix

Adds the prefix specified in the *Process* text box to the new signal name. For example, you have an existing signal *WIRE1*. If you specify `new` in the Process text box and click the *Add Prefix* button, the prefix *new* would be added to the new signal name and the new signal name would be *NEWWIRE1*.

Add Suffix

Adds the suffix specified in the *Process* text box to the new signal name. For example, you have an existing signal *WIRE1*. If you specify `new` in the Process text box and click the *Add Suffix* button, the suffix *new* would be added to the new signal name and the new signal name would be *WIRE1NEW*.

Paste Special: Change Signal names

This dialog box lists the conflicting signals in the source and target designs and allows you to change signal names.

Old Signal Names	Lists the signal names in the copied part
New Signal Names	Lists the signal names that would be used while pasting parts. By default, these names are same as *Old Signal Names*. You can enter a new value for the conflicting signal names.

Import Design

Procedure

Use this dialog box to select block or sheet(s) of the selected project in Import Design dialog box into the current project.

Libraries	Displays the list of all the libraries included in the `cds.lib` file. **Note:** If you select a library that does not contain any blocks, then the Import Design dialog box would display blank columns. A message prompting non-availability of blocks in the library appears.
Retain Hard Packaging Information (sheets only)	Select this check box to bring hard package properties along with the sheet(s).
Import	Click to import the selected sheet or sheets into the current project.
View Sheet	Click to view the sheets in the sheet viewer, which is equivalent of opening Design Entry HDL in the read-only mode.
Reset	Click to clear all the selected sheets and blocks in the right pane of the Sheet Import dialog box.
View Log	Click to display the log file report for design import.
Close	Click to close the Import Design dialog box.

Import Design: Block Re-Import

The Import Design: Block Re-import dialog box matches all the blocks being imported with the existing blocks in the library and lists the blocks that exist in the library. You can select block(s) to be re-imported and click *Continue* to re-import the selected blocks.

Import Design: Source information

The Import Design: Source information dialog box appears when you select the *Import* button in the Import Design dialog box. This dialog box provides a summary of the sheets or blocks to be imported. You can select options to import the sheets or blocks at the appropriate location.

Insert Sheet(s) at Page	Specify the page number where you would like to insert the selected sheet. **Note:** If you enter a page number that does not exist, Design Entry HDL inserts the required number of blank pages to complete the import operation at the requested page number.
of <design_name>	Select the name of the design where you want to insert the selected sheet.
Import block(s) by <mode>	Specifies whether the blocks should be imported in Read/Only (R/O) mode or Read-Write (R/W) mode.
Import the hierarchical block(s) in the sheet(s) imported by <mode>	Specifies whether the blocks in the sheet(s) being imported should be imported in Read/Only (R/O) mode or Read-Write (R/W) mode.
Copy block(s) in library	Specifies the library where you want to copy the imported block.

Import Design: Signal Name Clash

This dialog box lists the conflicting signals in the source and target designs and allows you to change signal names.

Old Signal Names	Lists the signal names in the source design
New Signal Names	Lists the signal names in the target design. You can edit these values.
Process	Allows you to add the specified text as a prefix or suffix to the signal name. **Note:** The *Process* field becomes active when you select *New Signal Name* field(s).

Add Suffix

Adds the text entered in the *Process* field as suffix to the signal name specified in the selected *New Signal Name* field(s).

Add Prefix

Adds the text entered in the *Process* field as prefix to the signal name specified in the selected *New Signal Name* field(s).

View Design

Procedure

Use this dialog box to specify the `project.cpm` file of the project that you want to open in Read-Only mode for browsing.

Project File (.cpm)/ Library Paths file (.lib)

Specify the path to the appropriate project file (.`cpm`). You can also browse to the required file.

The Project list box displays the preferred list of projects saved in the `preferred_projects.txt` file and the projects accessed in the current session. You can quickly select an available project from the list.

Baseline

Procedure

Use this dialog box to save and baseline a schematic. This dialog box appears when you choose *File – Save Baseline*. The caption of the dialog box has the following syntax:

```
Baseline: <ProjectName> Current Baseline <Current Version Number>.
```

For example, if you are baselining the `IC_design.cpm` project for the first time, the caption of the dialog box will be: *Baseline: <IC_design> Current Baseline <0.0>*

New Version

This field lets you specify a new version number for the schematic. Depending on the radio button you select, the field may populate automatically or can be entered manually.

- Select the *Major Version* radio button if a design has undergone major changes such as addition or deletion of a block. A major version, by default, automatically increments the current version number by 1 (a whole number). For example, if the current version of the design is 2.0, selecting the *Major Version* radio button increments the new version number to 3.0.
- Select the *Minor Version* radio button if a design has undergone minor changes such as renaming of a component or changing the value of a physical property. Selecting the *Minor Version* radio button increases the current version number by the nearest decimal point, automatically. For example, if the current version is 2.0, selecting the *Minor Version* radio button increments the new version number to 2.1.
- Select the *Other* radio button to enter a desired version number for the schematic in the text box. The version number should have zero as the last decimal point (for example: 1.3.0 and 1.2.1.0). You cannot have version numbers such as 1.4.5 and 2.3.7.

User Comments

Contains any version-specific comments for the new version. A user can enter these comments. This field is optional.

Auto Generated Messages	Contains system-generated messages in a grid that describe the history of changes made in the schematic, till the current version.
Msg ID	Contains a system-generated message ID. Every message has a corresponding message ID.
User	Specifies the user name of the user who made the change.
Date	Specifies the date on which a message was added.
Msg Type	Specifies whether a change is major or minor.
Message Text	Contains the text of the system-generated message.
Add to History	Check this check box to add system-generated messages in the log files, along with user comments (if any).
Baseline	Saves the new version of the design.
Cancel	Closes the dialog box without baselining the design.
Help	Displays the online Help for the Baseline dialog box.

QuickPick Browser Window

Procedure

This window appears over the *Global Navigation* window, as soon as you choose a cell, part or block from the QuickPick toolbar.

Filters	Displays components that match specific search criteria available in a column header. The column headers -- **Cell Name, Description** -- can use the wildcard character, asterisk (*). * matches any text string.
Cell Name	Lists the name of a cell.
Description (available for parts and blocks)	Contains the description of a part or block. ■ For parts, the description includes the part name concatenated with comma-separated key properties; ■ For blocks, the description includes only the library name.

Advanced Search >>	Opens the Part Information Manager window.

QuickPick Setup Dialog Box

Procedure

This dialog box lets you specify the library cells to be listed under a specific component category for a design project.

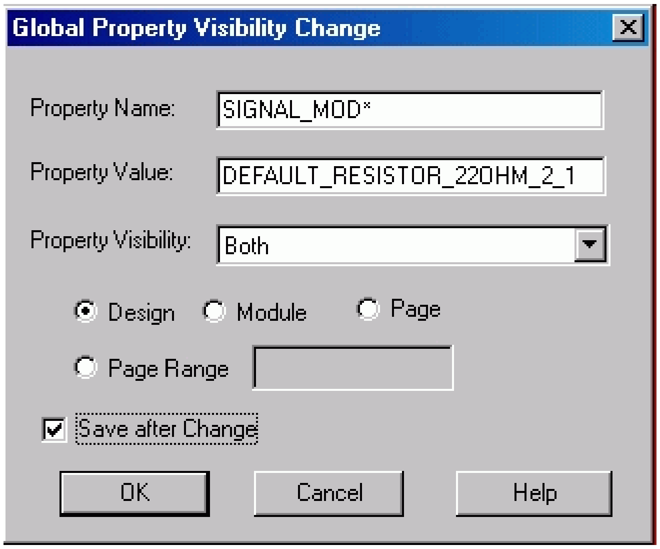
Use...	**To...**
cds.lib File	Specify the path to the `cds.lib` file of the design project.
Browse	Navigate to the required `cds.lib` file of the design project.
Setup File	Specify the path to an existing file containing the QuickPick Browser setup information.
Browse	Navigate to the required `.qps` file for the design project.
Icon	Choose a component category. You can choose from Power, Ground, Ports, Taps and Page Borders.
Tooltip	Enter the text to appear when you position the cursor over a component category.
Library	List all the libraries included in the `cds.lib` file.
Add Library	Add a complete library (all the cells) in the selected component category.
Remove Library	Remove a complete library (including all the cells) from the selected component category.
Available Cells	List all cells available in a library selected in the *Library* drop-down list.
Icon Cells	List all cells to be included in a selected component category (in the *Icon* drop-down list box) such as Power, Ground, Ports, Taps and Page Borders.
Add-->	Move a selected cell (listed in the *Available Cells* list) to the *Icon Cells* list.

Use...	To...
Add All-->>	Move all cells of the library listed in the *Available Cells* list to the *Icon Cells* list.
<--Remove	Remove the selected cell listed in the *Icon Cells* list to the *Available Cells* list.
<<--Remove All	Remove all cells of the library listed in the *Icon Cells* list to the *Available Cells* list.
Save	Save the setup information for the QuickPick Browser in the default project directory. The information is stored in the `qpsetup.qps` file at the CSF search location such as `cdssetup/concept` in the site area.
Save As	Save the setup information for the QuickPick Browser in a file and location, other than defaults.
OK	Close the dialog box and save the QuickPick Browser setup changes.
Cancel	Close the dialog box and cancel the QuickPick Browser setup changes.
Help	Lets you access the online Help for QuickPick Set up dialog box.

Global Property Visibility Change

Procedure

Use this dialog box to change the name/value visibility of a property globally.



Global Property Visibility Change

Property Name: SIGNAL_MOD*

Property Value: DEFAULT_RESISTOR_22OHM_2_1

Property Visibility: Both

● Design ○ Module ○ Page

○ Page Range

☑ Save after Change

Use...	**To...**
Property Name	Specify the name of the property whose visibility is to be changed.
Property Value	Specify the value that should match in order for the property to be displayed.
Property Visibility	■ Choose *Invisible* to make a property invisible on the schematic. ■ Choose *Name* to make only property names visible on the schematic without the associated property values. ■ Choose *Value* to make only property values visible on the schematic without the associated property names ■ Choose *Both* to make both property names and values visible on the schematic
Design	Choose this option to make the specified property visible in the entire design.

Use...	**To...**
Module	Choose this option to make the specified property visible in the current module or design.
Page	Choose this option to make the specified property visible in the current page.
Page Range	Choose this option to make the specified property visible in the specified page-range in the current design.
Save after change	Select this check box to save the changes to the design.

Model Import Wizard (Select Matching)

The Model Import wizard appears when you open a symbol in the symbol view and choose the Associate Model command from the PSpice Simulator menu or right-click and select Associate Model from the pop-up menu.

Use this control...	**To do this...**
Select library to pick matching models	Specify the path to the simulation library (`.lib`) containing the PSpice model to be associated with the selected Design Entry HDL part.
Matching Models	Display a list of PSpice models in the selected model library that can be associated with the Design Entry HDL symbol.
View Model Text	Display the model definition for the PSpice model currently selected from the Matching Models list.
Symbol pane	Display the name and the graphic for the Design Entry HDL symbol to which PSpice model is to be attached.
Next	Move to the next step.
Cancel	Cancel the process of associating a PSpice model to the Design Entry HDL symbol.

Model Import Wizard (Define Pin Mapping)

Use this page for pin to port mapping between the selected symbol shape and the model definition.

While you complete the pin-port mapping, you can view the symbol shape in the Symbol pane on the right of the wizard, and the use the View Model Text button to view the model definition.

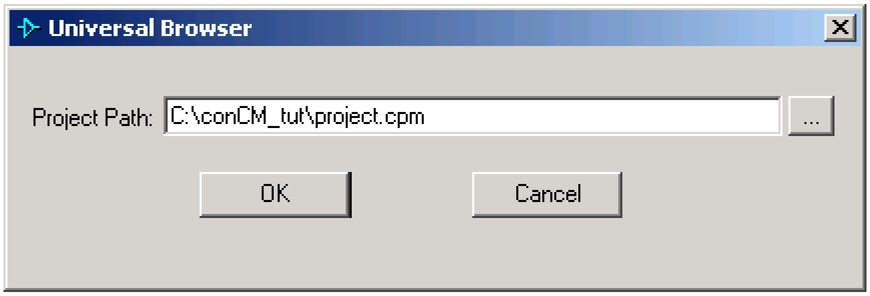
Note: All the symbol pins must be mapped to a model terminal. After you have mapped each symbol pin to a unique model terminal, if there are any optional model terminals left, you may leave them unmapped.

Control...	**Use...**
Model Terminal	Lists the port names from the model definition.

Control...	**Use...**
Symbol Pin	Lists the symbol pin names. From the drop-down list, select the pin name that is to be associated with the listed model terminal.
Optional Model Terminals	List the optional ports in the model definition. Depending on the availability of symbol pins, you may or may not have this option visible.
Symbol pane	Displays the shape of the open symbol to which PSpice model is being associated.
Back	Select this to move to the previous step, where you selected a matching symbol.
Cancel	Select this to cancel the process of associating an existing symbol to a simulation model.
Finish	Select this to complete the process of associating the selected PSpice Model to a the Design Entry HDL symbol and close the wizard.
View Model Text	Display the model text for the selected model in a new window.

Part Information Manager - Standalone

Use this dialog box to specify the path of the design project (`.cpm` file) that you want to open in Part Information Manager.



Project Path: C:\conCM_tut\project.cpm

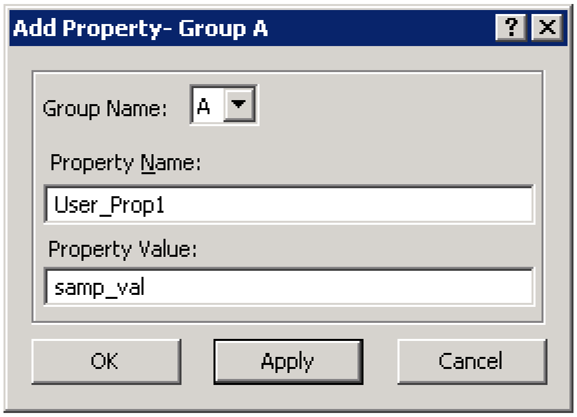
Use...	**To...**
Project Path	Specify the path to the appropriate project file (`.cpm`).

Browse button	Navigate to the location of the project file (.`cpm`).
OK	Close the dialog box and open the specified project in Part Information Manager.
Cancel	Close the dialog box and cancel the opening of the specified project in Part Information Manager.

Add Property Window

Procedure

Use this dialog box to add user properties to a groups of components. This functionality was earlier possible only using the `auto prop` command. You can open the Add Property window from the *Group – Add Property* menu. As with the `auto prop` command, the properties are automatically annotated onto the components within the group.



Group Name: A

Property Name: User_Prop1

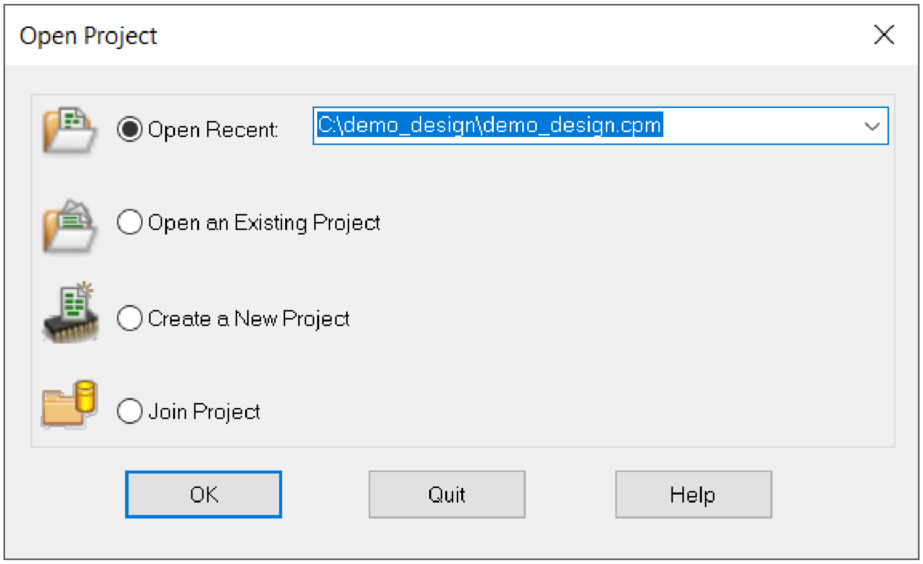
Property Value: samp_val

Use...	**To...**
Group Name	To specify the name of the group. You need to create a group before assigning properties to the group using the Add Property window.
Property Name	Name of the property to be added to the group.
Property Value	Value of the property to be assigned to the specified group.

Open Project

Use the Open Project dialog box to:

- open a recently accessed project
- open an existing project
- create a new project using the New Project wizard
- join a design project that is enabled for team design



Open Project

● Open Recent: C:\demo_design\demo_design.cpm

○ Open an Existing Project

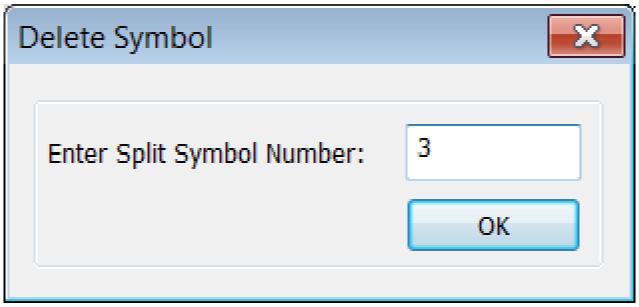
○ Create a New Project

○ Join Project

Distribute Port

Use this dialog to distribute ports across split symbols.

Use...	To...
Auto Distribute icon	Distributes ports across split symbols based on the settings specified in the Design Entry HDL Options-Split Symbol dialog.
Settings icon	Opens the Design Entry HDL Options-Split Symbol dialog.
Add Symbol icon	Adds a new split symbol to the hierarchical block.

Delete Symbol icon	Deletes the specified split symbol from the hierarchical block.  Delete Symbol Enter Split Symbol Number: 3
Reset icon	Resets the split symbols. All the split symbols for the hierarchical block are deleted.
Help icon	Opens the help page for this dialog.
Port Names	Displays the port name.
Retain Graphics All	Indicates that the symbol graphics will be retained as they are.
Symbol#	The number of the split symbol on which the port is present.
Customize Symbol Graphics	Opens the Customize Symbol Graphics dialog box, where you can modify the position and location of ports for symbols.

Customize Symbol Graphics

Use...	**To...**
Move Row Up icon	Moves a selected port name up in the Port Names list.
Move Row Down icon	Moves a selected port name down in the Port Names list.
Help icon	Opens the help page for this dialog.
Location	Displays the location of the port on the split symbol. The options are TOP, BOTTOM, LEFT, and RIGHT.
Type	Displays the type of port, whether input (I), output (O), or InOut (IO).

Port Names

Displays the port name.

Symbol#

The number of the split symbol on which the port is present.

Customize Symbol Graphics

Opens the Customize Symbol Graphics dialog box, where you can modify the position and location of ports for symbols.

Note: You can edit split symbols in two ways—by using the Customize Graphic Symbol dialog box or by opening the split symbol from the File – Open menu in the schematic canvas.

Important

> If you edit a split symbol in the Customize Graphic Symbol dialog box and later edit the same symbol in the View Open dialog box (*File – Open*), changes made in *View Open* may not reflect in *Customize Graphic Symbol.*

Select Variants

If you want to view the full list of variants in your schematic design, click *Variants*. This option is available under the Variants menu in Design Entry HDL for the following sub-menus:

- ❑ Edit Variant
- ❑ Remove Variant
- ❑ View Variant Schematic
- ❑ Mark for Variant(s)

Note: A CPM directive, `VAR_MENU_COUNT`, in the `START CONCEPTHDL` section allows you to control the number of variants that will be displayed in the Variants menu. The maximum number of variants that can be displayed is 20 and the minimum is 1. By default, lif the number of variants in your design exceeds ten, the sub-menu lists the Variants option.

Use...	To...

Select All	Click this check box if you want to select all the variants for deletion or if you want to mark the components you selected in the schematic canvas to these variants. Note that the Select All box is not enabled when editing a variant.
Help icon	Opens the help page for this dialog.

Edit NetGroup Membership

Use this dialog to add new members or to remove existing members from the net group.

Option...	**Use...**
NetGroup Name	Name of the net group being edited. This field is not enabled by editing.
Filter	Lists the net objects that can be added to the net group. The values in the drop-down list are: ■ Nets ■ Buses ■ NetGroups
Design Objects	Lists all available net objects of the type selected in the Filter drop-down list. **Note:** Only the net objects that are available in the design and are not added as member objects to any other net group are listed in this list
Name	Use this filter box to filter the net objects displayed in the Design Objects list by name.
NetGroup	An entry in tis column indicates the members removed from the net group in the current session.

Option...	Use...
NetGroup Member Objects	Lists the existing members of the net group. **Note:** Members added to the netgroup in the current session do not have an entry in the NetGroup column.

9

Naming Rules and Conventions

Overview of Naming Rules and Conventions

This section specifies naming rules and conventions used in Design Entry HDL and related tools. It also provides guidelines to specify names that will not have conflicts or adverse effects in the front-to-back flow.

Naming Rules in Design Entry HDL

When you specify names for project, design, signal, components, and properties in Design Entry HDL, you need to follow certain rules. You also follow certain conventions to ensure consistency and to eliminate errors when you transfer your design to Packager-XL or any other tool.

For details about name rules and conventions, refer to the different topics in this section. The following table describes the naming rules:

Objects	Naming Rules	Invalid Characters	Example	Non - Examples
Project Name	■ Use only lowercase letters, digits, and the underscore (_). ■ Use the extension `.cpm`.	All special characters other than the underscore.	`my_project`	`My_project`
Data Folder name	■ Use only letters, digits and the underscore	`\/:*?"<>|%;$[]'!`	`archive_7`	`archive%7`

Objects	Naming Rules	Invalid Characters	Example	Non - Examples
Library Name	■ Use only lowercase letters, numbers, and underscore. Remember that the `cds.lib` file, where you specify library names for a project, is case sensitive for pathnames. In all other cases, cds.lib is insensitive to case. For example DEFINE and define mean the same in a cds.lib file. However, d:\designs and D:\DESIGNS are two different paths.)		`my_lib`	`My_lib`
Design Name	■ Use only alphanumeric characters. ■ Use _ (underscore) or - (hyphen) ■ Use only lowercase letters. On the Windows platform, it is possible to use an uppercase design name. ■ Do not use spaces. Spaces in block names, root design and lower-level blocks, are not supported in the front-to-back flow.	Characters other than alphanumeric, _ (underscore) or - (hyphen) are not supported in design names.	`design1` `top2`	`design@1` `TOP2`

Objects	Naming Rules	Invalid Characters	Example	Non - Examples
Signal Name	■ Use only letters, numbers, and the following characters: ., -(hyphen), #, $, %, +, and `. ■ Use only positive integers as LSB and MSB for vector signals. (Remember that < and > can only appear for vector signals). **Note:** When the MULTI_FORMAT directive is set to ‘ON’ in the .cpm file, do not use 'space' and parenthesis for naming signals.	; ~ ` " ! **Note:** Spaces should not be used.	INTER_SIG*, new<2..0>_N	INTER~, new<2..0>_p
Property Name	■ Use only alphabets, digits, dollar ($), and the underscore. (A property name starting with $ specifies a soft property.) ■ Start with a letter. ■ Use a maximum of 31 characters.	! @ # % ^ & * ( ) -(hyphen) + = ~ ` { } [ ] \ | : ; " ` < > ,(comma) . ? / leading or trailing space	prop_name, $prop1_name	prop=name

Objects	Naming Rules	Invalid Characters	Example	Non - Examples
Property Value	■ Use alphanumeric characters and special characters except the listed invalid characters. ■ Use a maximum of 255 characters in Design Entry HDL **Note:** For PCB Editor, this limit is up to 1023 characters.	`! "``	`234, AB100`	`!234`
Primitive Name Or Part Name (as it appears in `chips.prt`)	■ Use only letters, digits, and the underscore. ■ Use a maximum of 30 characters.	~ ` ! * ( ) - (hyphen) + = | \ { }[ ] : ; " ` < > . ? space	`my_name123`	`!my_name`

Objects	Naming Rules	Invalid Characters	Example	Non - Examples
Pin Name	■ Use only letters, numbers, and the following characters: -(hyphen) # $ % + / = | ? ^ _(underscore). (and) (Although you can use (and) in pin names in Design Entry HDL by turning off multi_format_vector, hlibftb reports error.) ■ Follow low asserted pin names with an * or _N. Do not use any other nomenclature to specify assertion. ■ Use a maximum of 30 characters. ■ Use angular brackets only for sizeable or vector pins with a positive integer LSB and MSB. (Remember that vector or sizeable pin can only be added to a symbol or package and not in the Global Pin section)	; ! < > : \ " , (comma) * (Remember that * can only appear at the end of a pin name to specify a low assertion.)	new-pin	new\pin

Objects	Naming Rules	Invalid Characters	Example	Non - Examples
Pin Number	■ Use alphanumeric character including the underscore. ■ Use a maximum of 31 characters.	\ " !& @ ~ ` ^ < > . , (comma) : ; { }	23, p_n_20	p@n
Package Name	■ Use letters, digits, - (minus) and _ (underscore). ■ Use a maximum of 27 characters.		new_pack-er23	new+pack
Net Name	■ Use only printable characters. ■ Use a maximum of 31 characters.	! ‘	n_2@p	n!_2@p
Cell Name (The BODY_NAME property in chips.prt)	■ Use only letters, digits, and the following special characters: @ # - % ^ & () [] {} = / | , ? ~ ! * + _ ; > . ` embedded space(s) ■ Use a maximum of Max 2047 characters. ■ Leading spaces are not allowed.	\ : ' " < leading space(s)	p_t123 part.123 p@rt*1 p-t123	p123 p<t part:123’
Model Name/ Mapfile Name	■ Use the extensions .v or .vhd.	: / \ < > ? " |	myfile.v, myfile.vhd	my|file.v, myfile.txt
Symbol Name	■ Use only names of the form, sym_n where n is a non-zero positive integer.		sym_20	sym12

Note: Remember that `*` can be used only at the end of a signal name. A few other characters with special meaning can also be used. See Characters with Special Significance in Signal Names on page 537“.

Important

In Design Entry HDL, all property names and values are changed to uppercase. Unmark Upper-Case Input check box in the Text page of the Options dialog box to retain casing. You can also edit the cdsprop.paf file to specify the default behavior of specific properties. Use the keywords 'preservename' and ‘uppercase’ to indicate that the case of a property name and value should be preserved. Note that case-sensitivity is supported for schematic properties, opf properties, and properties in ppt and chips files.

Naming Conventions in Design Entry HDL

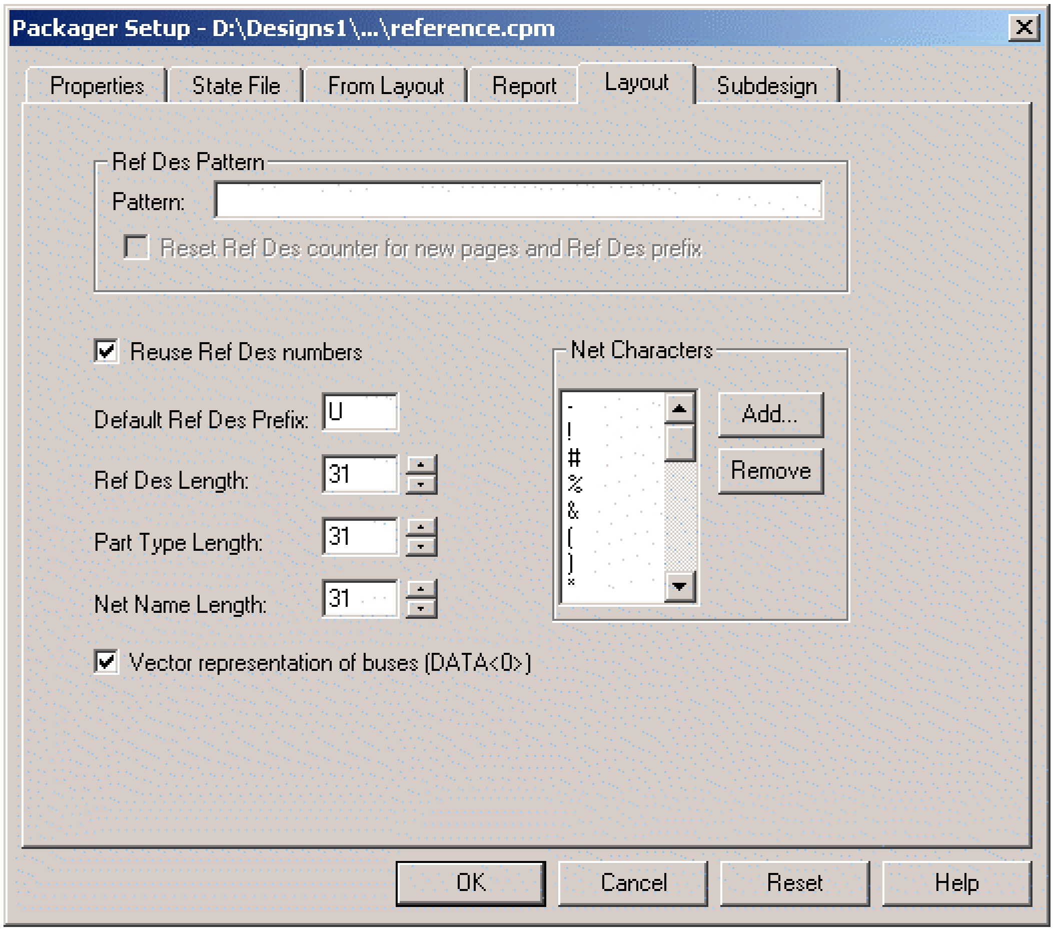
You can follow certain conventions to ensure consistency and avoid errors or conflicts when you transfer your designs to other tools. In addition, certain names have special meaning as described in the section Characters with Special Significance in Signal Names on page 537.

Some of the naming conventions are mentioned below:

Name	Convention
Project Name	Start the name with a letter. It is a good practice to keep the same name for the project and the directory containing the project.
Design Name	Use only lowercase letters, numbers, and the underscore. Start with a letter.
Signal Name	Start names with a letter and avoid verilog and VHDL keywords. Remember that Design Entry HDL is not case sensitive. As a result, Design Entry HDL treats two names that differ only in case as the same name. You should also avoid the following characters: `;  ` + "`" + ` ~ , "` (The semicolon, back quote, and comma cause Packager-XL to fail.)

Passing Signal Names to PCB Editor

When signals are passed to the Packager-XL tool, the signal name is changed or retained according to the `NET_NAME_CHARS` Packager-XL directive. You can set this directive from the Layout page of the Packager Setup dialog box as shown in the following figure:



Packager Setup - D:\Designs1\...\reference.cpm

Ref Des Pattern

Pattern:

□ Reset Ref Des counter for new pages and Ref Des prefix

☑ Reuse Ref Des numbers

Default Ref Des Prefix: U

Ref Des Length: 31

Part Type Length: 31

Net Name Length: 31

☑ Vector representation of buses (DATA<0>)

Net Characters

```
-
!
#
%
&
(
)
*
```

Note: To open the Packager Setup dialog box, from the Project Manager, select Design Sync followed by Export Physical and then click the Advanced button in the Export Physical dialog box.

You need to include characters that PCB Editor should retain. Characters not specified in the `NET_NAME_CHARS` directive are ignored by PCB Editor. Alphabetical and numeric characters are passed to Packager-XL irrespective of their inclusion in `NET_NAME_CHARS`. However, if a name starts with a digit, which is not a part of `NET_NAME_CHARS`, the net name will be prefixed with character A. In addition, netlister translates all characters to uppercase. Therefore, two signals that are different in Design Entry HDL because of differences in case are merged in Packager-XL.

Characters with Special Significance in Signal Names

You should use certain characters cautiously because they might alter the meaning of the names. Following are symbols that have special significance when used with a signal name:

Symbol	Description
:	Used to concatenate signals, and to specify the range and step size of a bus. While concatenating signals, a colon needs an operand on both sides. For example, A:B is legal, while AB: is illegal.
&	Represents concatenation and needs an operand on both sides when the MULTI_FORMAT directive is set to ‘ON’. When the MULTI_FORMAT directive is ‘OFF’, the ampersand character has no special meaning and can be used anywhere in a signal name.
`	Used for concatenating signals. To use a comma for concatenating signal names, ensure that the MULTI_FORMAT directive is set to ‘ON’. It needs an operand on both sides. For example, A,B is legal, while AB is illegal.
\G	Used for declaring the name of the signal as the “base” signal name for all its aliases or synonyms. The name of the base signal becomes the name of corresponding physical net in PCB Editor. A signal is its own base signal if it is not aliased to any other signal or if it is selected as the base signal.
\I	Used for Interface. Implies the signal is a port. Cadence recommends that you use the port symbols INPORT, OUTPORT, or INOUT instead of the \I suffix. The \I suffix declares ports as INOUT ports. If you use the port symbols, you can explicitly declare a port as an IN, OUT or INOUT port.
\L	Used for local.
/	Used at the beginning of a signal name to indicate a global signal.
!	Used at the beginning of a signal name to indicate a global signal.

Symbol	Description
`< >`	Indicates that the signal is a bus. The angle brackets must be matched correctly and must contain either a parameter or an integer. Must be the at the end of a name and cannot be used anywhere else in a signal name. Can only be followed by either `*` or `_N` to indicate a low asserted signal.
`( )`	Indicates that the signal is a bus when the MULTI_FORMAT directive is 'ON'. Must be matched correctly and contain either a parameter or an integer. When the MULTI_FORMAT directive is 'OFF', the parentheses have no special meaning and can be used anywhere in the signal name.
`[ ]`	Indicate that the signal is a bus when the MULTI_FORMAT directive is 'ON'. Must be matched correctly and contain either a parameter or an integer. When the MULTI_FORMAT directive is 'OFF', the square brackets have no special meaning and can be used anywhere in the signal name.
`*`	Indicates that the signal or pin is low-asserted when used at the end of a signal name or a pin name. Cadence recommends that you use the `_N` suffix to indicate a low-asserted signal or pin. A signal or pin is not low-asserted by default.
`0`	Is converted to `ZERO` in the netlist. The name `ZERO` indicates a low signal.
`1`	Is converted to `ONE` in the netlist. The name `ONE` indicates a high signal.
`_N`	Indicates that the signal or pin is low-asserted when used at the end of a signal name or a pin name. For example, `PRESET_N` or `OUT_N<7..0>`. A signal or pin is not low-asserted by default.

Caution

Remember that _N and * are the only characters that you can put after a subscript range. Therefore, `output<7..0>_N` ***or*** `output<7..0>*` ***are legal names but*** `output<7..0>N`***,*** `output<7..0>_*, output<7..0>_a` ***or*** `output<7..0>b` ***are not legal names.***

Step Size in Signal Names and Limitations

You use bit subscripts to specify number of bits that a signal represents and identify the bits. For example, the syntax `<bit1..bit2:step>` specifies a sub-range of bits beginning with `bit2` (LSB) and including every bit that is `step` bits apart up to `bit1` (MSB).

Following are a few limitations of step size in signal names:

- You cannot use the syntax `<bit1..bit2:step>` for interface signals because interface signals become pins if you create a symbol from a schematic.
- You cannot use the Design Entry SCALD syntax `<bit:width>`.
- You cannot use `<bit1..bit2:step>` for PATH properties.
- You cannot perform Global Find or Global Navigate operation on buses with step sizes in their names.
- You cannot tap from a bus that has step size in signal name.
- You cannot connect an unnamed vector signal to a pin because Design Entry HDL treats each bit of a signal of the form `NC<b1..b2> as NC.`

Name Mapping

Name mapping is a naming convention that makes data interoperable among Cadence® tools. Each Cadence tool has a name space that defines the rules for creating legal name for that tool. Name mapping allows these data to be transferred between the different tools.

The following table describes the name space related to Design Entry HDL.

Name Space	Normal Names	Escaped Names
Concept	This name space is not case sensitive and allows all printable ASCII characters except the apostrophe (‘), the quotation mark ("), the angular bracket (<), and the colon (:).	This name space allows quoted and escaped identifiers. Quoted identifiers are enclosed within apostrophes or quotation marks. The opening quote must be matched by the same symbol. You can use all printable ASCII characters including the angular brackets, space, and colon in quoted identifiers. To use a quotation mark in a quoted identifier, you must double the quote symbol or use a different pair to enclose the identifier. For example, to specify an identifier ab’c, you must write "ab’c" or ‘ab’’c’. Although you can use space in a quoted identifier, a space cannot be the first or last character. Escaped identifiers can contain all printable graphic ASCII characters enclosed between the backslash characters. To include a backslash character in an identifier, precede it by a backslash. For example, to specify an identifier ab\c, write \ab\\c\.

10

Commands and Switches Reference

This document describes the commands and switches to run Project Manager, Design Entry HDL, and related tools, such as Archiver, Rules Checker, Cross Referencer, and so on.

Note: All the switches that are displayed within square brackets are optional while other switches such as `-proj` are mandatory.

- archcore
- archopen
- bomhdl
- checkplus
- checkplusui
- cpmaccess
- creferhdl
- concepthdl
- ds
- hpfhdl
- libaccess
- partmgr
- projmgr
- psetup
- publishpdf
- vedit

archcore

You can create an archive of a design from the command prompt by using the `archcore` utility.

Command Switch

```
archcore -proj <project_name> -path <output_path> [-alldesigns] [-f
<file_name>] [-compresscmd <compress command>] [-delarchivedir] [-
ignorefile <list of files separated by white spaces>] [-views
<view1> <view2>…]
```

Description

`-proj`	Specifies the name of the project file `.cpm` to be archived.
`-path <output_path>`	Specifies the path of the output directory where the design is to be archived.
`-alldesigns`	Specifies that all root designs should be archived. By default, Archiver traverses only the root design specified in the `.cpm` file. If all root designs need to be archived, the `-alldesigns` option needs to be specified.
`-views <view1> <view2>…`	Specifies the views to traverse for archiving the design. You can specify the view as any combination of Schematic, Verilog, or VHDL. If you do not specify any view, Archiver traverses all views. **Note:** `-views` must be the last argument in the `archcore` command switch.
`-f <file_name>`	Specifies the name of the file that contains the list of directories and files to be archived along with the design. The `file_name` should specify the fully qualified path of the file.

`-compresscmd <compress command`

Specifies the compress command to be used in order to compress the archive. Specify the compresscmd parameter as:

“`cdszip -r $archive $location`”

This parameter must be enclosed within double-quotes.

`-delarchivedir`

Indicates that the archive directory should be deleted after creating the tar file

`[-ignorefile <list of files separated by white spaces>]`

Specifies the files, which need to be ignored while archiving. This parameter only accepts filenames enclosed within double-quotes and separated by spaces. It does not accept full paths to files.
Any occurrence of such file names in the project being archived are not archived. You can also use wildcard characters to specify filenames.

archopen

Use the `archopen` utility to launch the Archiver dialog box where you specify the archive to be opened.

Command Switch

```
archopen -proj <project_name>
```

Description

`-proj`	Specifies the name of the project file `.cpm`.

bomhdl

Use the `bomhdl` utility to generate a BOM report from the command line.

Command Switch

```
bomhdl.exe -proj <Project_File_Name> [-nographic]  [-t
<Template_File_Name>] [-f HTML | SS|TEXT] [-delim <delimeter>] [-o
<outfile name>] [-a YES | NO]  [-var <Variant_Database> [variant1
variant2...] [COMPARE] [ALL]]
```

To generate a BOM report from the command-line prompt, you provide inputs to the BOM-HDL tool by passing different switches. All the switches that are displayed within brackets are optional while other switches such as -proj are mandatory.

Description

`-nographic`

Specifies that BOM-HDL will run in the silent (nographic) mode (where no dialog boxes are displayed). The BOM report will be generated based on the settings that you define using the different switches or, if no particular switch is defined, then the BOM report will be generated using the settings stored in the project file.

Note: You will not get any message of successful completion. You can check the output in the location specified by the `-o` switch.

`-proj`

Specifies the path of the project file (`.cpm` file name).

Note: You need to specify the full path to the project file. An incomplete path or a wrong path will generate the error message, 'Either -proj switch absent or incorrect project path name'.

Note: You can specify the relative path from the current directory.

-t or -T <*template_file_name*>

[Required if you are working in the silent mode and the template file is not present in the default directories]

Specifies the name of the template file.

Depending on whether you specify the full or relative path of the template file, BOM-HDL searches the template file at the specified location and loads the file. If the file is not present at the specified location, then BOM-HDL searches for the template file in the `bom` view. Therefore, if your template file is present in the `bom` view, specify only the file name; otherwise, specify the complete path to the template file.

If the template file is not found in the specified path or if the *<Template_File_Name>* is not specified, then BOM-HDL searches for the locations as specified in the `setup.loc` file. It searches locations in the order specified and loads the template file from the location it finds first.

If the `-t` or `-T` switch is not used then BOM-HDL first searches for the template file settings in the project file. If no information for the template file is stored in the project file then BOM-HDL searches for locations as specified in the `setup.loc` file and loads the template file from the location it finds first.

-o or -O *<output_file name>* [optional]	Defines the name of the output BOM report file. In the graphic mode, BOM-HDL by default picks the value of the *Output File* field from the project (`.cpm`) file. If the project file does not have information specified about the output file, then the *Output File* field displays: ■ The `BOM.rpt` file in the `bom` view of the design as the default output filename if the report format is text file or spreadsheet ■ The `BOM.html` file in the `bom` view of the design as the default output filename if the report format is HTML. Depending on the report format type you choose and whether you are generating the BOM report for the base schematic or a variant, the name of the file in the *Output File* field automatically changes. For example, if you are generating a variant BOM report for the `INDIA` variant in the HTML format, then the name of the file will be `INDIA.html`. If you specify an output file but do not specify its path, then BOM-HDL creates the output file in the `bom` view of the current project. If the path information is available, then that information is used. **Note:** The value of the output file specified in the graphics mode will be seeded in the project file. In the silent (nographic) mode, BOM-HDL will generate the BOM report but will not seed the output file path in the project file.
-f [TEXT][HTML][SS] [optional]	Specifies the format of the BOM output file. `HTML` signifies the HTML file format, while `SS` denotes the spreadsheet format. The default format is ASCII text type. This is the format assigned when the `HTML` or `SS` format is not specified or no other formatting option is specified in the project file.
[-delim <delimeter>]	Specifies the delimiter to be used in report generation. <delimiter> can contain comma (,), semi-colon (;), colon (:), space ( ), period (.), hash (#), and (tab) as special characters, and must be mentioned without brackets.
-a [YES|NO] [optional]	Specifies whether or not filters be applied to BOM reports.

-var <Variant_Database> [variant_name] [ALL] [COMPARE] [optional]	Specifies the path of the variant database filename. If the path of the variant database is not specified, BOM-HDL searches for the variant file in the variant view of the selected design. You can also provide the name of the variant for which the BOM report is to be generated. Alternatively, you can use the ALL switch to generate BOM report for all variants or the COMPARE switch to generate the variant comparison BOM report. If you use multiple switches in the bomhdl command, then ensure that the -var switch is used as the last switch.

checkplus

Use the `checkplus` utility to run Rules Checker to check your design.

Command Switch

```
checkplus -proj <project_name> [-compiledfiledir <directory name>] [-
max_messages <num] [-I <include path>] [-r rule_file [rule_name]...]
```

Description

`-proj`	Specifies the project file.
`-compiledfiledir`	Specifies the directory where compiles rule files should go.
`-max_message <number>`	Specifies the maximum number of messages to be reported in a single run of Rules Checker.
`-i`	Specifies an include path. If the option is not specified, then rundir and installation default include paths are included.
`-r rule_file [rule_name]...`	Lets you specify a compiled rule (or rules) not included in your cp.dat file.

checkplusui

Use the `checkplusui` utility to invoke the stand-alone Rules Checker graphical user interface (GUI) (interactive mode).

Command Switch

```
checkplusui -proj <project_name>
```

Description

`-proj`	Specifies the project file.

cpmaccess

Use the `cpmaccess` utility to access the project's cpm file for reading, writing, or deleting directives from the command prompt.

Command Switch

```
cpmaccess -read <cpm_file> [program_name [directive_name]]]

OR
-write <cpm_file> program_name directive_name directive_value]

OR
-delete <cpm_file> program_name [directive_name]
```

Description

`-read`	Specifies that the given cpm file is to be read. If you specify the `program_name` and the `directive_name`, this command will return the value of the directive.
`-write`	Specifies that the value for a specific directive is to be written or modified in the given cpm file.
`-delete`	Specifies that the a specific directive is to be deleted from the given cpm file.
`cpm_file`	Specifies the name of the project file (`.cpm`).
`program_name`	Specifies the name of the program, for example if you want to access (for reading, writing, or deleting) a directive for the *DESIGNSYNC* program (under the `START_DESIGNSYNC` section), specify `DESIGNSYNC` as the program_name

`directive_name`

Specifies the name of the directive to be accessed. For example, to read the value of the *REPLACE_SYMBOL* directive in the `START_DESIGNSYNC` section, specify the value as `REPLACE_SYMBOL.`

`directive_value`

Specifies the value you want to set for a directive. For example, to set the value of `REPLACE_SYMBOL`, specify `1` or `0`.

creferhdl

Use the `creferhdl` utility to create cross-references for a design.

Note: Before you cross-reference a design from the command-line prompt, set all cross referencing options using the Cross Referencer Options dialog box. You can cross-reference a schematic from the command-line prompt. However, it is recommended that you use the CRefer dialog box to define all cross-referencing settings, and then, if required, use the command-line prompt to cross-reference the design.

Important

> If you want flat cross references to be generated for nets in hierarchical blocks that have been instantiated multiple times, or have been instantiated using split hierarchical symbols, you must make sure that you add offpage symbols to the nets.

Command Switch

```
creferhdl -proj <project_name> [-d] [-e] [-i] [-expand][-l] [-o] [-q]
[-p] [-proj] [-r][-s] [-z]
```

Description

`-proj`	Specifies the path to the project file you want to cross-reference.
`-d`	Deletes all the existing cross-references in the design.
`-e`	Retains duplicate entries.
`-i`	Omits input output arrows in cross-references
`-expand`	Creates a separate view for the current run.
`-l`	Puts block names in hierarchical cross-references.
`-o`	Sorts by page number only.
`-p`	Cross-references - flag body/wire spacing, in1/200 inch
`-q`	Cross-references - Xref spacing, in 1/200 of an inch

`-r`	Redoes all Xrefs.
`-s`	Scale text, multiply the default test size by this.
`-z`	Omit zone information from cross-references

concepthdl

Use the `concepthdl` utility to launch Design Entry HDL from the command line.

Command Switch

```
concepthdl -proj <project_name>
```

Description

`-proj`	Specifies the path to the project file you want to open.

ds

Use the `ds` utility to launch export or import physical commands from the command line.

Command Switch

```
ds -dlg <export|import> -proj<project_name>
```

Description

`-dlg <export|import>`	Specifies whether to run Export Physical or Import Physical.
`-proj`	Specifies the path to the project file you want to run the utility on.

hpfhdl

Use the `hpfhdl` utility to plot drawings in the HPF plotting mode from the command line.

Command Switch

```
hpfhdl [-f|-v outputfile] [-o] [-p papersize] [-2 <headerfile>
<path_to_drawing>]
```

Description

`-f`	Writes the data to a new version of the output file
`-v`	Writes a vector format file to a new version of the output file
`otputfile`	no output file is specified, the output is sent to the printer or the plotter
`-o`	This parameter operates if you specify the -f option, which implies that hpfhdl overwrites the new version of the output file (-f specification) rather than append data to the file specified, which is the default behavior.
`-p`	Specifies the paper size. This value must already be defined for the plotter in the `.cdsplotinit` file. The default is the first paper size entry defined for the plotter in the `.cdsplotinit` file.
`headerfile`	The name of the header file.
`path_to_drawing`	The path to the drawing you want to plot.

libaccess

Use the `libaccess` utility to access the cds.lib file from the command prompt.

Command Switch

```
libaccess <path_to_cds.lib>
-path <library_name>[<view_name>[file_name>]]]
OR
- libs
OR
-cells <library_name>
OR
-views <library_name> <cell_name>
OR
-files <library_name> <cell_name> <view_name>
OR
-find <cell_name>
OR
-exists <library_name>[<view_name>[file_name>]]]
```

Description

`<path_to_cds.lib>`	Specifies the path to the `cds.lib` file for the project.
`-path`	path to the library name, or the view name/file name.
`-libs`	Lists all the libraries referenced in the project.
`-cells`	Lists all the cells in the specified library
`-views`	Lists all the views in the given cell of the specified library.
`-files`	Lists all the files in the specified view of the given cell of the specified library.
`-find`	Finds the library in which the specified cell is found.

`-exists`	Returns whether the specified cell exists in the given library.

partmgr

Use the `partmgr` utility to launch Part Manager from the command line.

Command Switch

```
partmgr -proj <project_name> -product Concept_HDL_expert
```

Description

`-proj <project name>`	Specifies the path to the project file you want to run the utility on.
`-product`	The license with which you want to run the Part Manager application.

projmgr

Use the `projmgr` utility to launch Project Manager from the command line.

Command Switch

```
projmgr -proj <project name>
```

Description

`-proj <project name>`	Specifies the path to the project file you want to run the utility on.

psetup

Use the `psetup` utility to launch the Project Setup dialog box from the command line.

Command Switch

```
psetup -proj <project name>
```

Description

`-proj <project name>`	Specifies the path to the project file you want to run the utility on.

publishpdf

Use the `publishpdf` utility to create a PDF document of a given design from the command line. You can use the `publishpdf -help` switch to walk through all the options.

Command Switch

```
publishpdf -proj <project_name> [-product] [-cdslib] [-lib] [-cell]
[-view] [-page] [-all] [-range] [-active] [-print] -file
<destination_path> [-extscrpt] -variant <variant name or names
separated by a comma> [-unit] [-w] [-h] [-lmargin] [-rmargin] [-
tmargin] [-bmargin] [-scale] [-fitpage] [-landscape] [-portrait] [-
single] [-double] [-extscrpt] [-noprogress] [-bnw] [-pdfa]
```

Description

`-proj <project name>`	Specifies the name of the source project's .cpm file.
`-product <product license>`	Specifies the product license.
`-cdslib`	Specifies the cds.lib file
`-lib`	Specifies the library name
`-cell`	Specifies the cell name
`-view`	Specifies the view name
`-page`	Specifies the page name
`-all`	Publishes all the pages present in the current design
`-range`	Publishes a specified range of pages. If more than one page parameters are specified, the parameter specified last is honored.
`-active`	Publishes only the current page
`-print`	Publishes a printable black and white PDF

`-file`

Specifies the path to the directory where the published PDF should be saved. This is a mandatory parameter.

`-variant`

Specifies the name of the variant that should be published to a PDF file. Can be a list of variant names separated by a comma.

`-unit`

Specifies the unit in which you want to define the PDF page size. Can be inch or millimeter.

`-w`

Defines the width of the PDF in inches or mils. If 0, the PDF defaults to auto size.

`-h`

Defines the height of the PDF in inches or mils.

`-lmargin`

Defines the left margin of the published PDF.

`-rmargin`

Defines the right margin of the published PDF.

`-tmargin`

Defines the top margin of the published PDF.

`-bmargin`

Defines the bottom margin of the published PDF.

`-scale`

Specifies the scale of the published PDF.

`-fitpage`

Specifies whether the PDF should be fit to the page. If this switch is defined, the scale is ignored.

`-landscape`

Specifies that the PDF should be in landscape mode.

`-portrait`

Specifies that the PDF should be in portrait mode.

`-single`

Specifies a numeric value for single width wire thickness.

`-double`

Specifies a numeric value for double width wire thickness.

`-extscrpt`

Specifies the path of the xml file that contains Javascript to insert in the published PDF.

`-noprogress`	Suppresses the progress indicator dots on the command line.
`-bnw`	Generates a black and white PDF.
`-pdfa`	Generates a PDF that is compliant with the PDF/A standard.

vedit

Use the `vedit` utility to invoke the Variant Editor application.

Command Switch

```
vedit -proj <project_name>
```

Description

`-proj`	Specifies the project file for which Variant Editor needs to be launched. **Note:** If the `<project_name>` does not exists in the current directory, Variant Editor opens the File browser, which allows you to browse to the project file. If the `<project_name>` exists in the current directory, Variant Editor opens the project.

11

Managing Environment Variables

Overview of Environment Variables

Environment variables are strings that contain information about the working environment of a system and control the behavior of various programs. Environment variables contain information to determine where specific software is located or where to place files such as temporary files.

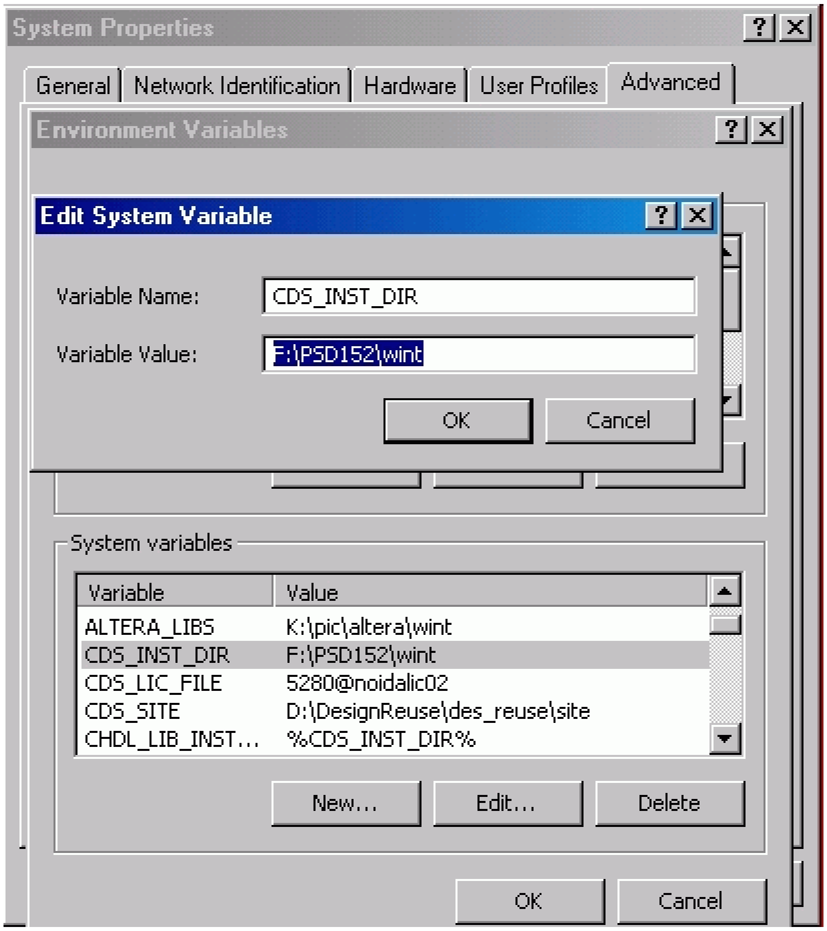
When you work with any tool, you need to set environment variables to specify the operating conditions at local and site levels. Most of the environment variables and the required path to different executables are set on your system by the installer. However, there are some other variables that you still need to set. This document describes the environment variables required to run Design Entry HDL and related tools and how to set these variables.

Setting Environment Variables on Windows Platform

To set environment variables on Windows platform, perform the following steps:

1. Right-click *My Computer* and then click *Properties*.
2. Click the *Advanced* tab.

3. Click *Environment Variables*.



Variable Name: CDS_INST_DIR

Variable Value: F:\PSD152\wint

Variable	Value
ALTERA_LIBS	K:\pic\altera\wint
CDS_INST_DIR	F:\PSD152\wint
CDS_LIC_FILE	5280@noidalic02
CDS_SITE	D:\DesignReuse\des_reuse\site
CHDL_LIB_INST...	%CDS_INST_DIR%

4. Click one the following options, for either a user or a system variable:
   - ❑ Click *New* to add a new variable name and value.
   - ❑ Click an existing variable, and then click *Edit* to change its name or value.
   - ❑ Click an existing variable, and then click *Delete* to remove it.
5. Specify variable name and variable value and click *OK*.
6. Close the *Environment Variables* and the *System Properties* dialog boxes

Common Environment Variables

- ■ Global Environment Variables
  - ❑ Library Path Variables

- ❑ System Variables
- ❑ Installed Variables

- ■ User-Defined Variables

Some of the common environment variables that you need to set for Design Entry HDL and other tools are described in this section.

Environment Variables	Description	Example
`ALLEGRO_MWUSER_DIR`	Sets the location of the registry.	
`AUTO_PLUMBING`	Set this variable to disable the auto-detection of global plumbing bodies.	
`CDS_HPF_TMP`	This environment variable points at the location of the spool file for HPF plotting. In HPF plotting, the spool file `vw.spool` is created in a temporary directory (`/tmp`) of the system from where Design Entry HDL was launched. To create the spool file in another location, you need to set the environment variable `CDS_HPF_TMP` to a directory where you have write permissions. **Note:** The spool file vw.spool is created in the directory specified using the `CDS_HPF_TMP` environment variable only if you are plotting a schematic for which you have read-only permissions. If you set this environment variable and plot a schematic for which you have write permissions, the spool file `vw.spool` is created in the project directory.	

Environment Variables	Description	Example
CDS_IGNORE_LIC_FEATURE	Set this environment variable to TRUE to bypass the validation of feature strings in the license file by the tools. After setting this variable, you will see all the product licenses in the Cadence Product Choices dialog box and you can select the product according to the license you have.	
CDS_LIC_FILE	This environment variable is set to specify the license server for Cadence tool(s).	setenv CDS_LIC_FILE 5280@noidalic02
CDS_SITE	This is a location environment variable, which specifies the location of the site project file.	setenv CDS_SITE /user_name/site
CDS_TEXT_EDITOR	Set this environment variable to set the default text editor for Design Entry HDL. If this variable is set, Design Entry HDL will not use the text editor specified in the Tools tab of the Project Setup dialog box. You should unset this variable if you want Design Entry HDL to use the text editor as specified in the Tools tab of the Project Setup dialog.	
CDS_VARIANT_PROP_VIS - {BOTH | NAME | VALUE | INVISIBLE}	Set this environment variable to control the display of variant properties in the schematic backannotated by Variant Editor.	

Environment Variables	Description	Example
`CDSROOT`	This environment variable points to the directory where you have installed Cadence tool(s).	If you install the tool at /cdsinst/ spb22.1, the value of this variable is set as `CDSROOT/ cdsinst/ spb22.1` To change the path, use the following command: `setenv CDSROOT/ cdsinst/ spb22.1`
`CONCEPT_DESCEND_EDIT_LIST`	Set this environment variable to descend into drawings while in Hierarchy mode. If you have `vlog_rtl, sch_1`, and `sym_1` views of the drawing and wish to descend into them when you double-click the top level drawing, you need to set the following environment variable. `Setenv CONCEPT_DESCEND_EDIT_LIST vlog_rtl, sch_1, sym_1`	
`DISABLE_VIEW_REPORTS_DIALOG`	Set this environment variable to suppress the report viewing dialog box before using the `bomhdl` command.	

Environment Variables	Description	Example
`GDM_MAIL_HOSTNAME`	Set this variable to specify the mail server name on Windows platform.	
`GENERATE_WRAPPER`	Set this variable for backward compatibility with the 13.5 release, if you want the Design Entry HDL Uprev utility to process wrappers.	
`LD_LIBRARY_PATH`	This environment variable is used as a list of directories to search for the .dll files to load.	
`MAX_NETS_IN_XNETS`	Controls the maximum number of nets allowed in an XNet.	`MAX_NETS_IN_XNET '50'`
	To update the new value of this directive automatically in the Constraint Manager database every time you launch DE-HDL or System Capture, use `REFRESH_` as a prefix with the directive.	`REFRESH_MAX_NETS_IN_XNET '50'`
	Set the value to any number greater than 0.	
`MWUSER_DIRECTORY`	Specifies the full path to the user's Windows directory.	
`PATH`	This environment variable is used as a list of directories to search for various executable files to be run.	

Environment Variables	Description	Example
PDFFONT	Set this environment variable to fix text alignment issues in a PDF document of a schematic. This environment variable stretches the height and width of a text object to align it properly as it appears in the actual design. The range of values you can assign to this variable is between 1 and 4. Republishing a design after setting this variables fixes the text alignment issues.	`PDFFONT = 1.65`
REFRESH_MAX_NETS_IN_XNET	See *MAX_NETS_IN_XNETS*	
SHLIB_PATH	This environment variable points to the location of shared libraries on an HP_UX system.	
TEMP/TMP	Set this variable to specify the location to place temporary files created by the tool.	`setenv TEMP c:\cadence\temp`

TZ	Set this environment variable to set the current time zone.	`TZ = GMT`
		`TZ = PST`
		`TZ = PDT`

12

Troubleshooting Allegro Design Entry HDL

This appendix lists the solutions to some common or intermittent problems encountered while using Allegro Design Entry HDL.

- Using mixed case names and spaces in design names causes problems in front-to-back flow, especially with CRefer.
- Unable to launch Constraint Manager from specific design in Allegro Design Entry HDL
- Incorrect property definitions causing Netrev errors while importing a netlist in the board
- Genview fails with an error on a missing entity.tmp file.
- Packager-XL is crashing when running from Design Entry HDL in Solaris system
- Part Developer crashes when any cell in any library is opened
- Design Entry HDL generates a Dr. Watson error when parts with component property values greater than 255 characters are added and saved.
- CRefer crashes if a large number of NC pins are added to a part.
- Unable to start Design Entry HDL after reinstalling it due to an operating system crash
- Packager-XL crashes generating Windows Program Error window
- Corrupted CRefer properties may cause Design Entry HDL to crash when particular schematic page is opened in Design Entry HDL.
- Packager-XL crashes at the end of packaging step

Using mixed case names and spaces in design names causes problems in front-to-back flow, especially with CRefer.

Example

I have created a project schematic that appears to be correct. When I save the schematic, no errors or warnings are issued. When I run Export Physical, a netlist is created successfully. However, when I run CRefer, I see the following error written to the `cref.log` file.

```
CRefer fails with error "Cell: libname.%cellname is not present Cell:
proj_lib.%MyDesign is not present.
```

The project `cds.lib` file defined as `proj_lib` as my project worklib and the design is named `My_Design`.

Solution

Change the design name from `MyDesign` to `mydesign`. Launch the Project Setup dialog box of Project Manager, and change the project design name (the `DESIGN_NAME` directive) from `MyDesign` to `mydesign`. This should allow CRefer to run and design to be processed across the front-to-back flow.

Note: It is recommended that you use lower case names and do not use spaces for defining design names and cell names.

Unable to launch Constraint Manager from specific design in Allegro Design Entry HDL

Description

When you launch Constraint Manager from few designs in Design Entry HDL, the Constraint Manager window may appear briefly and then disappear or may not appear at all. The following MPS error is returned by Design Entry HDL.

```
MPS Error: MPSC: bad Handle: 0xbe52e0
```

Reason

It is possible that you have placed instances of a component in the schematic in Design Entry HDL and have not yet added any wires to connect them.

Solution

Ensure that component pins are connected with wires and launch Constraint Manager again from Design Entry HDL. It should run.

Incorrect property definitions causing Netrev errors while importing a netlist in the board

Description:

You may find the following Netrev (305) error while importing a netlist in a template board file in Allegro. This error will occur for every component in the netlist.

```
ERROR (305) Device/Symbol check error detected..... 'Attribute Definitions are
incompatible for attribute ....
```

Reason

This problem may occur if

- there is an improperly defined user-defined property in the Allegro board file to which the netlist is being imported.
- footprints are not upreved to the latest version.

Solution

To verify bad property definition:

1. Choose *Setup – Property Definition* from the main Allegro PCB Editor menu.

   The errant property definition is displayed in the *Available Properties* pane of the Define User Properties dialog box.

2. Select the property.

   The Data Elements and Data Type of the property are displayed in the right hand pane of the Define User Properties dialog box.

To correct the problem:

➤ Select at least one data type for the errant property and *Apply* the change.

You should now be able to import the netlist successfully.

Genview fails with an error on a missing entity.tmp file.

Description

If you have created a schematic project in a folder which has a folder above it containing spaces, then Genview may fail with an error message about not finding the `entity.tmp` file.

Reason

One possible cause might be that the folder above the project directory contains spaces in the folder names.

Solution

Move the project to a folder with no spaces in the path. Genview should complete successfully.

Packager-XL is crashing when running from Design Entry HDL in Solaris system

Description

While Packager-XL will run from the command line in Solaris systems, it may crash when run from Design Entry HDL in following situations:

Possible Reasons

1. The correct values of the view names are not set up correctly in the Project Setup dialog box.
2. The opf directory is corrupt.
3. The symbol.css files (page borders) do not contain the following property assignment:

   ```
   COMMENT_BODY = TRUE
   ```

Solution

Setting correct values of view names in the Project Setup box

1. Launch Project Manager and open Project Setup by clicking the Setup icon.

2. In the Project Setup dialog box, under the Views tab, make sure the following values are set for the corresponding view names:

   Packaged: packaged

   Part Table: part_table

3. If either of the above statements is missing, enter the correct values as mentioned above and repackage the design.

Fixing the opf directory

1. If you have a flat design, try deleting or renaming the opf directory to `opf.old`. The opf directory is present within the design folder.
2. Reload the design in the Project Manager.
3. Open Design Entry HDL and run Export Physical.
4. Verify if the crash is resolved.

Assigning COMMENT_BODY property in the symbol.css file

Check if the page borders (the `symbol.css` files) used in the design have the `COMMENT_BODY = TRUE` property added onto them. If the property is not present in the `symbol.css` file:

1. Add `COMMENT_BODY = TRUE` in the `symbol.css` file
2. Save the design and then run Export Physical again.

Part Developer crashes when any cell in any library is opened

Description

Part Developer (PDV) may crash when any cell in any library is opened if there is an incorrectly configured user customized site setup (`CDS_SITE`).

Solution

To confirm whether an incorrectly configured user customized site setup is causing Part Developer to crash, rename the `CDS_SITE` environment variable and restart the Part Developer.

If this fixes the problem, check specifically any custom entries in $`CDS_SITE/cdssetup/LMAN`.

Design Entry HDL generates a Dr. Watson error when parts with component property values greater than 255 characters are added and saved.

Description

If you assign properties with values greater than 255 characters to parts in a design and save the design, then Design Entry HDL may crash and generates the following Dr. Watson error.

```
Error: file is corrupt - unknown keyword 0x203
```

Reason

This crash is caused by one or more parts that you added to the design. If a part has a property attached to it and the value of the property has more than 255 characters, then Design Entry HDL will crash when you open the design.

Solution

To prevent the crash ensure that all component property values are 255 characters or less.

e.

CRefer crashes if a large number of NC pins are added to a part.

Reason

There is a limit of 8000 on the number of `NC_PINS` that you can add to a part as a property.

Solution

It is recommended that you split the symbol and import the NC pins from the package and make a new Design Entry HDL symbol version for them. You need not connect these symbols to anything, just attach them as logical pins.

Unable to start Design Entry HDL after reinstalling it due to an operating system crash

Reason

One of the possible reasons could be that the operating system variable `PATH` has stale or extraneous sub-paths pointing to other installations of Cadence software.

Solution

1. Edit the `PATH`. In the *Advanced* tab of the *Settings – Control Panel – System* window, click *Environment Variables*.
2. Ensure that *only* the currently valid Cadence installation is included in the `PATH`.

Packager-XL crashes generating Windows Program Error window

Description

While generating netlist, Packager-XL may fail generating a Windows Program Error window informing that the `pxl.exe` file has generated errors and is being closed by Windows and that an error log is being created. The crash may seem unexplained and `pxl.log` file would not be generated.

Reason

This is an extremely rare error. One of the likely causes of this error is that one or more of the parts used in the design may have a `chips.prt` file without any primitive defined. For example, the file might have the following content:

```
FILE_TYPE=LIBRARY_PARTS;
TIME=' Created/Modified on Thu Sep 30 15:18:21 2004' ;
END.
```

Solution

If the problem has occurred only recently and the project has been successfully packaged before, then the likely culprit would be a part that you have recently added to the design. You may also want to check with your library developer and check the content of the parts that have been recently changed in the reference libraries.

Corrupted CRefer properties may cause Design Entry HDL to crash when particular schematic page is opened in Design Entry HDL.

Description

This is a rare situation when corrupted CRefer properties may cause Design Entry HDL to crash when particular schematic page is opened pageX in Design Entry HDL. A Dr. Watson log file was generated and you may get error like one displayed below.

```
no property at (-2704 2775) to attach property to
```

Reason

The error message points to a location on the design page that crashes (pageX) when opened in Design Entry HDL.

To determine the cause of the problem:

1. Perform a search for the coordinates listed in the `pageX.csa` file.

   If the coordinates point to a CRefer property, then CRefer is causing the design corruption.

2. Restore the design from the backup and open the problem page. Observe all property assignments by checking the `PROPERTY_NAME=PROPERTY_VALUE` pairs. You may probably find culprit properties to be attached to an XREF rather than a net.

Tip

Use *Display – Attachment* to view property assignments.

Solution

To resolve the problem, you can

- Re-attach the culprit properties to the correct net, save the page, and re-run CRefer. You can run *Display – Attachments* to verify that the property is assigned to the correct net.

or

- move or delete the `pageX.csb` (binary) file. Design Entry HDL will then open the `pageX.csa` (ASCII) file instead, and you can then perform the troubleshooting steps described above.

Packager-XL crashes at the end of packaging step

Description

Packager-XL may crash at the end of the packaging step when you have parts in design with physical part name length greater than the limit set in Packager Setup dialog box.

Solution

You can increase the limit for *Part type Length* in Layout tab of project setup dialog box. The default limit is 31, and you can increase it up to 255.

13

Glossary

assertion level

Part of a signal name, it describes the active state of the signal when asserted. By convention, a signal is active high for positive logic and active low for negative logic. An * represents active low - for example, RESET* is an active low signal. Two signals with the same name but different assertion levels are not the same signal.

attribute

Information that Design Entry HDL lets you attach to objects (components, wires/nets, and pins) in a schematic. Attribute information is passed to other design programs for processing. An attribute consists of a name-value pair. Attributes are also called *properties*. See also *constraint*.

attribute file

A file that contains properties, their associated values, and some display information. Because different types of objects (components, wires, and pins) have different properties associated with them, they need to have different attribute files. A good way to add several properties to an object and ensure their names and values are correct is to use an attribute file as a template. See also *attribute*.

automatic routing

A Design Entry HDL function that automatically routes wires (*Wire – Route*) around objects in a schematic.

backannotate

The process of updating a Design Entry HDL schematic with information on new parts, connectivity, and properties from the Design Synchronization and Design Association tools. Usually, you backannotate the design after the first error-free run of Design Synchronization and then again after the design has been processed by a physical design system.

block

A hierarchical representation of a logical collection that can be reused in a schematic.

body

The symbolic representation of a component or design block. This is now called *symbol*.

BODY drawing (symbol)

The symbolic representation of a library component that you add to your design. This drawing defines the shape, pins, and general properties of the library component.

bubble pins

Low-asserted pins represented by circles on pins and indicated with a low-asserted signal name (*).

bus tap

Tapping a subset of signals from a bus. See also *tap.*

bus-through pins

Special pins placed on a component to make it easier to wire a group of components together. Bus-through pins have the same name as the corresponding visible pin.

To find out if a component has bus-through pins, you can use *Display – Pins* to display an asterisk at every pin location.

C-tap body

The default bus tap provided in the Design Entry HDL standard library.

category

Refers to a group of components arranged hierarchically.

cds.lib

A file containing library definitions.

cell

Software representation of a component. Consider a cell to be a collection of views that describe an individual building block of a chip or a system.

chips.prt

A file containing physical information about a component.

component

Refers to the logical characteristics of a library part.

Part Information Manager

A dialog box in Design Entry HDL that lists active libraries and their contents, both drawings and components.

component instance

The placement of a component one or more times on a schematic.

configuration

A collection of views that control how a design is compiled and simulated.

connectivity design data file

A file that defines how all the components and nets connect together logically. This file is used by Design Entry HDL to generate the resulting VHDL or Verilog.

constant signal

A signal that has a numeric name. For example, a signal named `123`.

See also, non-constant signal

constraint

A restriction on the physical implementation of a design object.

cross probe

The process of identifying corresponding parts, packages, and signals in the Design Entry HDL schematic and PCB Editor.

design

A schematic drawing created in Design Entry HDL.

DOC drawing

A drawing containing only graphics. DOC drawings are used for documentation purposes; no electrical or logical checks are done on them.

DRC

Design Rule Checking.

entity

The view of a cell that contains the definition, including port (pin) definitions, for the current drawing (cell). Several checks are made to ensure that entity declarations, symbols, and schematics are in agreement.

expand

To build a complete design including all levels of the hierarchy based on views specified in the current expansion configuration.

filter

Screens file names, markers, and so on in the current directory and lists only those that match the filter. An asterisk (*) or a blank field lists all the drawings or markers.

flat design

A design in which all parts of the drawing come from Design Entry HDL or user-defined libraries and are one-to-one logical representations of the physical parts. All of the interconnecting wiring within the design is entered pin-to-pin. Best suited for small designs that do not have sophisticated bus requirements or reuse portions of circuitry.

grid

Defines where wires and pins meet in the schematic. Design Entry HDL supports three grid types:

- ❑ Logic grid for schematic
- ❑ Symbol grid for symbol drawings
- ❑ Document grid for DOC drawings

hard property

Properties that you add to the schematic to specify packaging assignments. Hard properties are included in the connectivity files and thus also in the Verilog/VHDL netlist. They differ from soft properties, which are essentially documentation properties on the schematic and are not included in the netlist.

hierarchical design

A design that is organized into modules to reuse many of the same circuit functions and isolate portions of the design for teamwork assignments. Using a block design lets you refer to a collection of logic without having to include the logic in the drawing. Hierarchical blocks simplify a drawing. This is also called *block design*.

Hierarchy Editor

A tool to create and edit configurations, which can be used in netlisting. You can also view the components of your design hierarchy using this tool.

injected property

A property that appears to the right of a PPT format definition or part row. Packager-XL passes these properties to Allegro in the physical netlist, - for example, company-specific part numbers, costs, or package types.

interface signal

A signal property (\I) assigned to pins in block diagrams to indicate an interface signal from a higher level drawing. In a flat design, this is a signal in the schematic that corresponds to a pin in the symbol drawing.

key property

A property that appears to the left of a PPT format definition or part row. Packager-XL uses these properties to uniquely identify the physical part to use from the various table entries. For example, a resistor part table may use VALUE or TOLERANCE to select a specific physical part.

library

A collection of components from which you can select a component to place in a drawing.

library properties

Librarian-generated properties on symbols, chips, and in the Physical Part Table (PPT). Only the librarian can modify library properties.

marker

An error, warning, or information item that indicates a rule violation in your schematic. Markers are generated using the *Tools – Check* menu command, the CheckPlus utility, Design Synchronization, and Packager-XL.

net

A set of pins that are electrically connected.

netlist

An ASCII text file that describes the electrical connectivity (wires/nets and components) of a drawing.

non-constant signal

A signal that has an alphabetical or alphanumeric name. For example, `ADDRESS`, `DATA1`, `1CLOCK` and so on.

See also, constant signal

NOT body

Used to change the logic convention of a signal. If a signal is asserted low, it is considered to be a negative logic signal. If a signal is asserted high, it is considered to be a positive logic signal. The NOT body is used to change the logic convention of a signal without introducing an actual logical inversion. This implies the state of the signal is not changed, it is just considered to be of the opposite logic convention.

orthogonal

Bent to route around objects in a schematic. This is an alternative to direct (diagonal) placement.

package

(noun) In VHDL, a collection of types, constants, subprograms, and so on, usually intended to implement some particular service or to isolate a group of related items.

(verb) The process of translating a logical netlist into a physical netlist. Design Synchronization takes a logical representation of a schematic and applies the physical attributes necessary to allow physical layout.

page

Refers to a page in a design. If the amount of logic required to define a design does not fit on a single page, the drawing might extend to more than one page.

part

Refers to the physical symbol derived from the logical representation of a component or design block.

physical

Refers to the physical properties associated with a library component.

Physical Part Table (PPT)

Used to map logical parts in the schematic to physical parts for a layout.

pin

Conductors that protrude from packages. Pins allow the component to be connected logically to wires and other components in the logical design.

placeholder property

A temporary property assigned to the symbol drawing of a part. These properties serve as substitutes for part properties that will be assigned later in the schematic design. Placeholder properties let you predefine the location and text size of part properties through the part symbol drawing.

placeholder value

Substitutes a real property value. It is indicated with a **?** value.

PPT Browser

Lets you select parts based on the properties defined in the PPT file, such as company part number or preferred status.

primitive

The symbol name in the *chips.prt* file.

project

The work area for a design, including all the views of the design, links to libraries, and setup information such as Physical Part Table, configuration, and expansion directives. Separate directories exist for each design project.

property

A logical characteristic of a design object. It is information that Design Entry HDL lets you attach to objects (components, nets, and pins) in a schematic. Property information is passed to other design programs for processing. A property consists of a name-value pair. Properties are also called *attributes*. See also *constraint*.

ratsnest line

In a design drawing, a line that shows a logical connection between two pins, connect lines, or vias. Elements connected by the same ratsnest line are part of the same net. The ratsnest shows the circuit logic and, for ECL circuits, the order in which pins are to be connected.

reference designator

The designator, or identification code, for a component.

reference library

A library containing cells that describe common components potentially used in many designs.

root drawing

The top-level drawing in your design. This is the drawing that Design Entry HDL opens by default when you start an editing session.

route

To autoroute a wire (*Wire – Route*). This is an alternative to manually drawing a wire (*Wire –Draw*) around objects on a schematic.

rubberbanding

A feature of interactive commands in which the lines that are attached to an element of the design drawing “stretch’’ as you move the element with the mouse.

rules-driven design

User-defined design characteristics that can be specified by the schematic (as properties on components, pins, or nets) that are recognized by Allegro and determine processing results.

scalar signal

A signal having a single bit.

SCH drawing

A Design Entry HDL drawing that contains a *schematic*.

schematic

The standard type of drawing created with Design Entry HDL to represent the logic of components or design blocks that make up a circuit. The symbolic drawing is generated in a physical layout tool. A schematic can contain library components and design blocks that represent other schematics.

schematic properties

Modifiable properties that are defined when editing the schematic.

scope

You can assign one of three different scopes to a signal:

INTERFACE Used on signals you want to access from a higher level of a hierarchy. Represented by \I.

GLOBAL Used on signals that you need to access on all levels of hierarchy. Represented by \G.

LOCAL Indicates that the signal is recognized only at its own level. No special characters are required because the local scope is the default.

script files

Let you perform repetitive tasks in Design Entry HDL. You can build a script by editing a file and adding the commands in the sequence you want them to execute. You can use scripts to set up forms for routing, placing, and artwork or executing a series of check plots. Scripts can call other scripts.

section

Refers to a physical section on a logical component. Pin numbers are different in different sections of the component. You can section a component either before or after you package the design.

signal

Wire connections between components that support communication of dynamic data between components. Signals having the same name are interpreted as one signal; this is how signals are connected across multiple pages of a drawing.

signal bits

Signals can have a single bit (scalar signals) or multiple bits (vectored signals). The bit portion of the signal name is called the bit subscript and gives the bit information. Bit subscripts are enclosed in angle brackets, for example, <3..0>.

SKILL

A proprietary Cadence high-level interactive programming language based on the popular artificial intelligence language, LISP.

soft property

Properties that can change from one backannotation to the next. Soft properties are documentation properties on the schematic and are not included in the netlist. They differ from hard properties, which are included in the connectivity files and thus also in the Verilog/VHDL netlist.

structured design

Uses bus signals and memory and register depth. A structured design minimizes the number of interconnections and parts on the schematic.

swap

To exchange the locations of two logically identical pins within a function. This minimizes the average ratsnest crossings in a layout.

SYM drawing

A Design Entry HDL drawing that contains a *symbol*.

symbol

The symbolic representation of a library component that you add to your design. This drawing defines the shape, pins, and general properties of the library component.

symbol properties

Librarian-generated properties defined on a component through its symbol description and not by editing the schematic.

system properties

Non-modifiable schematic properties that Design Entry HDL adds.

tap body

Cadence-supplied taps found in the Design Entry HDL standard library: C-tap, tap.body, bustap.body, msbtap.body, and lsbtap.body.

text

Includes text can be signal names, properties, and notes.

user-defined net

A net with a signal name on it. Conversely, an unnamed net is one for which the user did not specify a net name and Design Entry HDL specifies the net name.

vectored signal

A signal having multiple bits.

version

Different graphical but functionally equivalent representations of a component, all of which refer to the same logic drawing. If the version is not specified, Design Entry HDL assumes the version to be1.

VHDL

VHSIC Hardware Description Language.

view

Designs are represented by these views in Design Entry HDL: schematic (or logic), symbol (or body), VHDL, and Verilog. Using *Tools – Generate View* in Design Entry HDL, you can generate one view of a design from another.

visibility

Refers to the amount of property or pin information displayed on a schematic.

wire

An electrical connection. A single wire can be an entire net, or, where there are many connections, a wire can be a segment of a net. This is also called *signal*.

wire orientation

Bent to route around objects in a schematic versus direct (diagonal).

Index