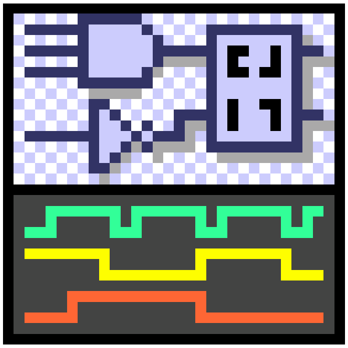
# *DesignWorks™ 4.7*

## *Macintosh® Version*
## *Users Guide*

July 2011



## IMPORTANT NOTICE

Capilano Computing Systems Ltd. (“Capilano”) retains all ownership rights to the DesignWorks™ program and all other software and documentation making up the DesignWorks package. Use of the DesignWorks software is governed by the license agreement accompanying the original media.

Your right to copy the DesignWorks software and this publication is limited by copyright law and your end user license agreement. Making copies, adaptations or compilation works (except copies for archival purposes or as an essential step in the utilization of the software) without prior written authorization of Capilano, is prohibited by law and constitutes a punishable violation of the law.

Capilano provides this publication “as is” without warranty of any kind, either express or implied, including but not limited to the implied warranties or conditions of merchantability or fitness for a particular purpose. In no event shall Capilano be liable for any loss of profits, loss of business, loss of use of data, interruption of business, or for indirect, special, incidental or consequential damages of any kind, even if Capilano has been advised of the possibility of such damages arising form any defect or error in this publication or in the DesignWorks software.

Capilano reserves the right to update this publication from time to time without notice. Some of the information in the publication refers to characteristics of third party products over which Capilano has no control. This information is provided for the convenience of DesignWorks users only and no warranty is made as to its correctness or timeliness.

Copyright ©2011 All rights reserved.

DesignWorks and LogicWorks are trademarks of Capilano Computing Systems Ltd. Other trademarks used in this publication are property of their respective owners.

Printed in Canada.

Capilano can be contacted at:

Capilano Computing
2631 Viking Way, Unit 218
Richmond, B.C., V6V 3B5
Canada

phone 604-522-6200

email info@capilano.com

WWW http://www.capilano.com

# Table of Contents

## Chapter 6—Before Starting a Major Design. . . . . 117

## Chapter 7—Device Packaging and Naming . . . . . 125

---

## Chapter 8—Tools for Searching, Browsing and Error Checking . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 155

---

## Chapter 9—Attributes . . . . . . . . . . . . . . . . . . . . . . . 173

## Chapter 12—Device Symbols. . . . . . . . . . . . . . . . . 271

# Introduction

# 1

Welcome to the DesignWorks™ schematic entry tool from Capilano Computing. DesignWorks is built with features designed to allow it to form the core of your electronics design system.

This chapter will point you to the resources you need to get started using the package as quickly as possible.

## Where to Start (First Time Users)

We suggest you ease yourself into the world of schematic editing with DesignWorks by taking the following steps:

**STEP 1**— Install using the standard Mac App store procedure.

**STEP 2**—Work first through the separate demonstration manual supplied with the package. It provides step-by-step instructions for basic schematic editing.

**STEP 3**—Refer to Chapter 6—Before Starting a Major Design on page 117 for information on choices that you should consider before investing too much work in a schematic.

**STEP 4**—Refer to Chapter 4—Basic Procedures on page 53 and Chapter 5—Basic Schematic Editing on page 69 for general schematic editing procedures.

As you work with DesignWorks you will have occasions to look for information in this manual. It is organized into a number of parts, sorted more or less in order of the depth and complexity of the material. Later parts address issues in larger designs, interfacing to other systems, customizing the package, etc.

If a question comes up that the manual doesn't answer, we are available for technical support by phone or on-line. See Appendix H—Technical Support on page 401.

## Where to Start (Upgrade Users)

This manual has been reorganized from earlier versions to be more procedure-oriented. We hope you will find it easier to locate clear instructions on specific issues of working with your designs.

The next section lists the major features in DesignWorks 4.x and where to find information on them. You may want to scan down the list and note any items that are relevant to your design work and look up the new information.

In any case, you should note these points before proceeding with any major design movement to version 4.x:

> There have been no significant changes in design and library file format between the 3.x and 4.x versions. A minor piece of default font information has been added to the design file format, so reading a file that was saved using 4.x back into a 3.x version will result in a "Unrecognized data has been skipped." warning. This can be ignored.
>
> There has been no major change in internal design structure in this version, so an schematic features that were used in version 3.x will still be available in 4.x without conversion.
>
> The report scripting language has undergone major revision in this version. 99% of scripts written for version 3.x should be compatible with the new version, but we recommend testing your netlist output before proceeding with a large design. Try generating the same netlist from both versions and compare the results with a text compare utility.
>
> The installation and usage of design templates, libraries, scripts, etc. has been consolidated in the new "design kit" organization. It is still possible to use the same file organization that was used in 3.x, but you may want to consider taking advantage of the new design kit features to customize your environment, improve error checking, etc.

# Schematic Features in 4.x

Here is a summary of new features in 4.x, relative to the 3.1.5 version.

## General Features

Intel native execution.

Major enhancements in scripting and report generation. These are discussed in more detail below.

The “design kit” concept to help you customize DesignWorks for your work. This will be most visible when you start the program and see the New Design dialog, but also affects library organization, netlisting, error checking, etc. See Chapter 13—Design Kits and Sheet Templates on page 335.

Productivity tools such as ConnMaker for generating connector symbols. See “Auto-creating Connector Symbols” on page 316.

## Schematic Editing Changes

Major changes in “New Design” dialog with customizable template list, access to examples, and clearer layout. See “Creating a New Design” on page 56.

Major reorganization of signal and device naming and packaging options to improve clarity. A different box is displayed depending on hierarchy mode. See “Enabling Naming and Packaging Options” on page 127.

Many improvements in Update from Lib:

> Update all devices, even in closed sub-circuits
>
> Update devices by type name, thus merging cases where multiple similar types were created accidentally.
>
> More attribute update options.

See “Updating a Symbol from a Library” on page 278.

Timed auto-backup and auto-checkpoint options. See “Backup Procedures” on page 64.

You can now control what type of objects are selected when you click or drag select items by holding the D, S, P, T or G keys on the keyboard. See “Restricting Object Types while Selecting” on page 73.

“Fit to Sheet” auto-scale printing option. See “Fitting the Diagram to the Available Paper” on page 62.

Major changes to Bus Pin Info dialog: You can now add and delete bus internal pins without using DevEditor, plus you can now join more than one pin to a single signal. See “Changing Bus Pin Connections” on page 215.

Improved bus, signal, pin and device info dialogs.

Better display of attribute values in the ? probe tool. Fields are sorted and null values are removed.

Non-flickering device placement.

## Gate Packaging Features

Elimination of 32-gate limit in Packager.

No name packaging option. If Name.Prefix is $NONAME, no name is assigned. “Specifying that a Device Should be Unnamed When Placed” on page 138.

Packager can be left enabled even with packaging errors.

Packager “free unit report” is now handled by report generation.

Improved packager error checking.

## Hierarchy Features

Issue a warning when editing an internal circuit with and external reference.

Save and restore instance data when updating internal circuits after a definition change.

A new auto-generated identifier called a “locator” which guarantees the unique identification of any object in a hierarchical design. This can be used for error location and other purposes.

Better checking for port/pin mismatch in hierarchical designs.

Improved access to internal circuits. If Find or ErrorScript locates an item in an internal circuit, it can be displayed immediately in most cases.

## Symbol Editing Features

DevEditor now allows pin settings to be changed on multiple selected pins. For example, for a 32-bit output bus, it is no longer necessary to set each pin individually. See “Entering Pin Information” on page 303.

Edit a symbol in place on the schematic. See “Editing a Device Symbol in a Schematic” on page 283.

Place Internal Circuit command auto-generates a symbol and attaches a sub-circuit in one operation. See “Placing a Subcircuit” on page 248.

Default attributes in a device symbol can now be set to Always Visible/Use Design's Default Visibility/Never Visible. See “Setting Part and Pin Attributes” on page 295.

## Major Report Generation/Scripting Enhancements

***The new report generation and scripting features are described in detail in a separate manual "DesignWorks Script Lanaguage Reference" provided in electronic form with the DesignWorks distribution.***

The Report Generator has been renamed “Scripter” to reflect its more general purpose nature. Many new script features have been added, while retaining 99% compatibility with old report scripts. Features are intended to allow customization of DesignWorks in the areas of error checking, report generation, simple back annotation, user prompting, etc. The features include:

Improved file handling: direct access to directory path names for important file locations.

Unix-style regular expressions allow powerful extraction and modification of data.

You can now read a line-oriented text file and extract data from it.

You can now directly generate Unix or DOS/Windows style text files.

Commands to put up simple prompt boxes to get user input.

Powerful date formatting commands with access to date created and modified.

Many structure commands: $IF/$ELSE, $WHILE, etc.

Translation tables.

Ability to generate a report to a text item on the schematic.

Ability to modify the "primary" vs. "secondary" status (and some other properties) of attribute field definitions from a report.

Ability to set the hierarchy mode of a design from a script.

More control over progress display

Ability to create a secondary "transcript" file at the same time, to output report logs or messages while generating the main report.

Generalization of keyword usage so that almost any keyword can be used anywhere it makes sense.

Simple arithmetic and comparison commands.

Text variables.

Find objects by complex expression, e.g. using regular expressions.

New function oriented syntax for most commands, i.e. arguments in parentheses.

Ability to set attribute values.

More options for inclusion of bus names in signal names.

Ability to select and object from within a script.

Ability to test and set OKErrors bits to implement a "Mark as OK" function in error checking.

You can set a value break to insert a new heading or page break when a sorted field value changes.

Create a directory and write multiple output files into it.

New $SYSTEMOPEN command sends a message to the Finder to open a file in another application.

Performance improvements.

## Other Tools

DesignKits tool to help with installing design kits. See "Choosing and Activating a Design Kit" on page 337.

ErrorScript tool replaces ErrorFind with a scriptable environment. See "Using the ErrorScript Tool" on page 166.

Connector Maker tool auto-generates connector symbols. This was resurrected from DesignWorks 2.5 by popular demand. See "Auto-creating Connector Symbols" on page 316.

Browser mode improvements: "Browse" mode is now truly read-only. Multi-cell cut/paste operations can be done in "Edit" mode. See "Selecting and Editing Cells" on page 162.

## Notes Regarding Copyright and Trademarks

The DesignWorks software and manual are copyrighted products. The software license you have purchased entitles you to use the software on a single machine, with copies being made only for backup purposes. Any unauthorized copying of the program or documentation is subject to prosecution.

A number of product trademarks are referred to in this manual. DesignWorks, LogicWorks, MEDA, and Modular Electronic Design Application are trademarks of Capilano Computing Systems Ltd. All other trademarks used are property of their respective holders.

# User Interface

# 2

This chapter provides general information on the use of windows, drawing tools and other user-interface features of DesignWorks.

## Mouse Button Usage

Three different mouse button actions are used for various functions in DesignWorks. For clarity we will use the following terminology when referring to these actions in the remainder of the manual:

<u>Click</u>—means press and release the button without moving the mouse. Example: To select a device, click on it.

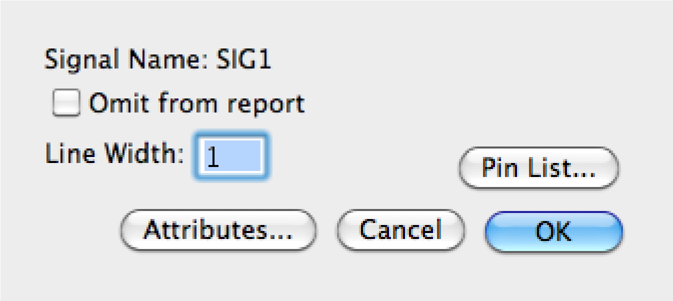
<u>Click and drag</u>—means press the mouse button and hold it pressed while moving the mouse to the appropriate position for the next action. Example: To move a device click and drag it to the desired new position.

<u>Double-click</u>—press and release the button twice in quick succession without moving the mouse. Example: To open a device's internal circuit, double-click on the device.

## Dialog Boxes

Many DesignWorks functions require information to be displayed or prompted from the user. To do this a special window called a “Dialog Box” is displayed, such as the following one which is used when a Get Info command is executed for a signal.

In general, the controls in dialog boxes will behave according to Macintosh standards, with the following exception: Since many of the boxes require text entry with multiple lines, the Return key <u>does not</u> normally cause the box to be closed and the default action to be performed; this function has been given to the Enter key. If the flashing text cursor is currently in a text box that allows multiple line entry, such as the attributes box, then the Return key will simply insert a carriage return in the text. If the insertion point is located in a text box that does not allow multiple lines, then the Return key will behave the same as the Tab key and cause the next text box to be activated.



Signal Name: SIG1

☐ Omit from report

Line Width: 1

In most dialog boxes requiring text entry, the keyboard equivalents for the clipboard command Cut (Command-X), Copy (Command-C) and Paste (Command-V) are active and can be used to transfer text to or from a text box.

## Window Usage

### Circuit Windows

Each circuit window displays one page of a circuit schematic. The title on a circuit window will be the name of the circuit file with the number of the page displayed in that window, as in “Interface Board:3”. Any number of pages in a given circuit and any number of circuits can be displayed simultaneously. At any given time only one page of one circuit is “current”, that being the one in the topmost window. Any other window can be made current simply by clicking the mouse anywhere in the window.

### Tool Windows

External modules, or Tools, can create their own windows which may be displayed concurrently with other DesignWorks windows. In general these obey the normal rules for resizing, etc., but their function is totally defined by the module.

## Closing a window

Clicking in the Go-Away box on a Circuit window has the effect of closing the circuit page. If the page being closed was the only one open on the circuit then the circuit is closed.

## Redisplaying a window

When a circuit page window is closed it is removed from the Window menu to avoid clutter. To reopen a page in the current circuit level, use the Pages command in the Drawing menu. To reopen an internal-circuit window, double-click on the device in question.

## Resizing a window

To enlarge or reduce a Circuit window, position the pointer in the size box at the lower right corner of the window and press and hold the mouse button. As long as the button is held down, a gray outline of the window will track the mouse movements. When the button is released the window will be redrawn with the new size and shape.

To expand the circuit window to fill the whole screen, click in the "Zoom" box at the upper right corner of the window. Clicking again in this box will cause the window to return to its original size.

## Moving a window

To move the Circuit window, position the pointer in the title bar anywhere along the top edge of the window and press and hold the mouse button. As long as the button is held down, a gray outline of the window will track the mouse movements. When the button is released the window will be redrawn at its new position.

## The Window Menu

The Window menu provides a means of bringing to the front any window currently open.

## Keyboard Usage

The Macintosh keyboard is only absolutely required when entering names for devices or signals, or for placing random text notations on the drawing. However, the command, shift and option keys on the keyboard can be used with many editing operations to invoke special features such as auto-alignment, auto-naming and different signal line drawing methods. In addition the arrow “cursor” keys (if available) can be used as a convenient way of setting symbol rotation while placing devices or pasting circuit groups. These options are described in detail in the relevant chapters.

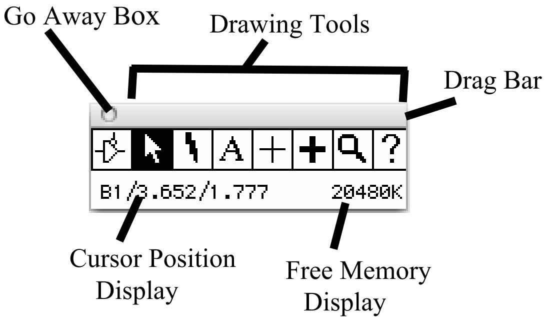
## Pop-up Menus

At any time while editing a diagram you can hold the control key and click on a schematic object. A pop-up menu will appear under the cursor allowing you to select from commands appropriate to that object. E.g. the menu for a device contains commands to get device information, edit attributes, open the internal circuit, flip or rotate the symbol, Cut and Copy operations, change gate package assignment, etc.

Separate pop-up menus are available for devices, signals, pins, attributes, and (if clicking in open space on the drawing) the circuit itself.

## Tool Palette and Status Display

The tool palette is a small window which always remains in front of any circuit windows currently open. The individual items in this display are described in their own sections below.



## Moving the Tool Palette

The tool palette can be moved to any desired position on the screen by clicking and dragging in its drag bar.

## Hiding the Tool Palette

The tool palette can be removed from the screen by clicking once in its “go-away” box. To re-display the palette, select the Show Tool Palette menu item in the Edit menu.

## Status Display

The lower area of the tool palette window is used to optionally display several items of status information.

### Cursor Position Display

The cursor position is shown as a grid reference and an X and Y distance in inches from the sheet origin (which may be any corner, depending on sheet setup). The cursor position display can be enabled or disabled in the Design Preferences command in the Drawing menu.

### Free Memory Display

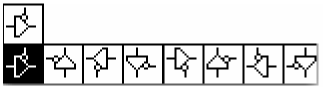
If enabled, the amount of free memory available for circuit data is displayed in the lower right corner of the tool palette window. This number is shown in Kbytes, i.e. multiples of 1024 bytes. The free memory display can be enabled or disabled in the Design Preferences command in the Drawing menu.

## Orientation Control

The orientation icon shows the orientation that will be used when the next device or circuit scrap is pasted into the schematic. The orientation can be changed by clicking on the icon, using the arrow keys on the keyboard, or by using the Orientation menu command.

### Orientation Pop-up Menu

Clicking and holding the orientation icon in the tool palette will cause the following pop-up menu to appear:



Choose any one of the 8 possible orientations by sliding across this menu. The selected orientation will be displayed in the tool palette icon.

This selection has no effect on items already placed in the schematic. It affects only the flickering image of items currently being placed and future devices selected from the parts palette.

***More information on symbol rotation is given in "Flipping and Rotating a Device" on page 85.***

## Schematic Drawing Tools

The seven drawing tools are used to set the cursor mode for schematic editing. More information on cursor modes is given in the following section.

## Tools Menu

The Tools menu provides direct access to the functional modules of DesignWorks. Tools can be added as new options are installed or special modules are created for specific applications. A number of utility modules are included with the release that may or may not be installed in your system.

The standard DesignWorks 4.x Schematic release contains these items in the Tools menu:

| | |
|---|---|
| **Back Annotation** | Back annotate from PCB System |
| **Browser** | Attribute display and editing |
| **Circuit to Library** | Copy devices from current circuit to the currently slected library |
| **Connector Maker** | Autocreate connector symbols |
| **Design Kits** | Activate and deactivate design kits |
| **Device Editor** | Create and edit device symbols |
| **Error Script** | Run error checking scripts |
| **Find** | Find devices and signals by name and attribute value |
| **Drawing** | Display a schematic window |
| **Library to Circuit** | Copy devices from currently selected library to the current circuit. |
| **Scripter** | Run a script for report generation or analysis. |

## Use of the Pointer or Cursor

In subsequent descriptions, we will refer to the small shape which tracks the mouse position on the screen as the “pointer” or “cursor”. In DesignWorks there are a number of different cursor modes used which determine what action will be performed when the mouse button is clicked. Following is a summary of the cursor modes. More detailed descriptions of operations performed in each mode are provided in later reference chapters.

Note that the cursor shape sometimes differs from the tool palette icon for ease of pointing.

| **Tool Palette Icon** | **Initial Cursor Shape** | **Equivalent Menu Command** | **Description** |
|---|---|---|---|
| ↖ | ↖ | Point | Used to select or drag objects, extend signals. |

| | | | |
|---|---|---|---|
| ✚ | ✖ | Draw Bus | Used to create a new bus line or extend an existing bus. Clicking once fixes a corner, double-clicking terminates the line. |
| + | × | Draw Sig | Used while creating a new signal line or extending an existing signal. Clicking once fixes a corner, double-clicking terminates the line. Note that most signal drawing operations can also be done in Point mode. |
| A | ✎ | Text | Used to select a signal or device to name, or to place random text on the diagram. Point at the item you want to name and press and hold the mouse button. Move to where you want the name to appear, then release the button. |
| ϟ | ϟ | Zap | Used to remove single objects. Press the button to remove whatever the tip of the cursor is pointing at. Objects can also be removed in groups by selecting them and using the Clear command or delete key. |
| 🔍 | 🔍 | Magnify | Used to zoom in and out. Clicking on a point or dragging down and right zooms in, dragging up and left zooms out. |
| ? | ? | Attribute Probe | When the “dot” portion of the cursor is clicked on a device, pin or signal, the contents of the primary attribute fields for that object are displayed. |

# Tutorial

# 3

The purpose of this tutorial chapter is to give you a quick overview of the steps in volved in schematic creation. You can then refer to the later reference chapters to find more information on specific topics of interest.

**TIP: We strongly suggest that you take a look at Chapter 6—Before Starting a Major Design on page 117 before proceeding with any significant design projects. A small amount of time spent on thinking through your design process now can save a lot of headaches later!**

To allow you to get a quick overview and then delve into specific topics of interest, this tutorial chapter is divided into these sections:

The 5-Minute Schematic Diagram -- *NOTE: All other sections below assume you have worked through this one!*

Advanced Schematic Editing

Creating Device Symbols and Hierarchical Design

Using DesignWorks with SPICE-based Simulators

## Manual Format

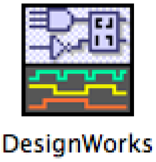
In all of the following sections, action instructions are shown in **bold face like this**, whereas explanatory text is shown in normal typeface.

## The 5-minute Schematic Diagram

In this first tutorial section we will show you how quick and easy it is to put together a complete schematic diagram and generate a netlist out for your board layout package.

**TIP: If you are new to the Macintosh or have any questions about the operation of the tools and control you see on the screen, you may want to refer to Chapter 2—User Interface on page 9 before proceeding with this section.**

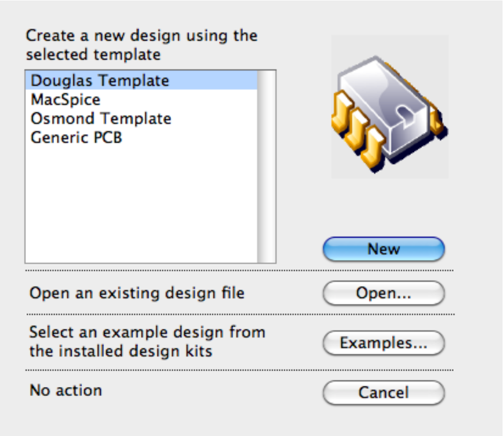
## Starting the Program



**If it is not already running, double-click on the DesignWorks™ icon to start the program.**

After a moment of loading the program and opening libraries, you will see the New Design dialog.

## Creating a New Design


Create a new design using the selected template

Douglas Template
MacSpice
Osmond Template
Generic PCB

Open an existing design file

Select an example design from the installed design kits

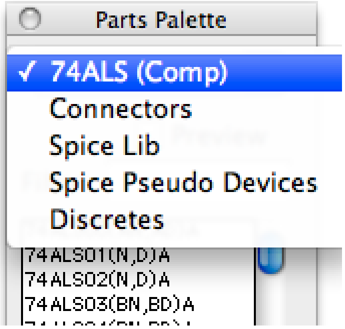
No action

The New Design box allows you to select a design mode for your application. Each selection enables the group of options most commonly used for the application type. All options can be changed later, if desired. The list of available design templates that appears here depends on which design kits are currently activated. If you do not see one listed that you need, refer to “Choosing and Activating a Design Kit” on page 337.

**Click on the "Generic PCB" button, if not already selected, then click on the New Design button.**

This selection gives us default settings suitable for a simple PCB design.

## Choosing a Library

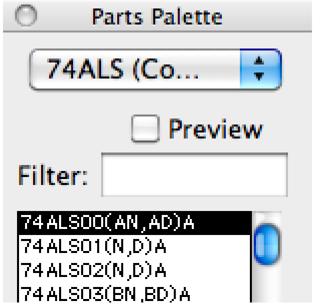


**Move the cursor to the library selection pop-up menu and choose the library "74ALS {Comp}".**

The contents of this library will now be displayed in the part selection list.

NOTE: The exact parts listing may vary from what is shown here, depending on the active design kits.

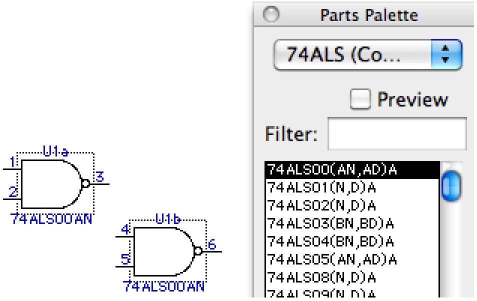
## Selecting a Part



**Move the cursor to the part selection list and double-click on the item "74ALS00...". Move the cursor into the schematic drawing area.**

You will see a flickering image of the selected part following the cursor movement. This part does not become a permanent part of the schematic until you click the mouse button.

## Placing a Device

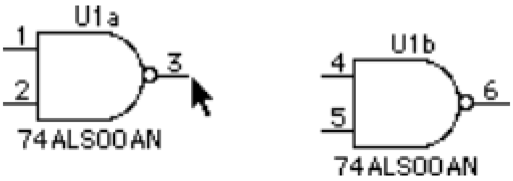


**Move the cursor to an area just left of the parts palette and click the mouse once to place a device in the schematic. Move and click again to place a second device as shown. Order of placement is not important.**

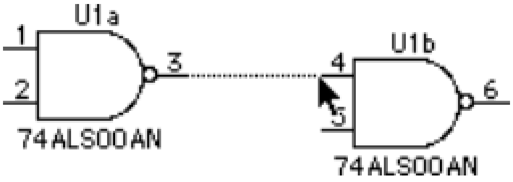
You can continue to place devices of the same type just by clicking in the desired locations. Notice that each device is automatically labeled with its part type and package assignment. This automatic assignment can be disabled if desired.

**Return to the normal pointer by typing the spacebar or escape character on your keyboard.**

## Wiring Pins



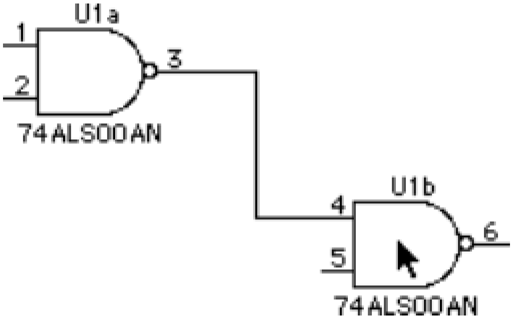
**Move the arrow cursor exactly to the end of the output pin on the left-hand device, as shown, then click <u>and hold</u> the mouse button.**



## Wiring Pins (cont'd)

**With the mouse button still held down, move to the right away from the output pin. Move so the arrow cursor is exactly positioned over the end of the upper input pin of the right-hand device, then release the mouse button.**

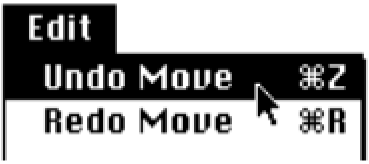
You will see the signal line flash briefly indicating that a connection has been made.



## Moving Devices

**Click and drag on the right-hand device and move it to a new position, as shown.**

Notice that the connected signal line stays attached and stretches to follow the device movements.



## Using Undo/Redo

- **Select the Undo command in the Edit menu.**
- **Select the Undo command again, and repeat until all items you have placed have disappeared.**
- **Now select the Redo command and repeat it until all edits are redone.**

Most schematic editing operations can be undone and redone up to 10 levels. Major structural changes (like adding a page or hierarchy level) or any operation involving a dialog box cannot be undone.

## Part Selection by Name

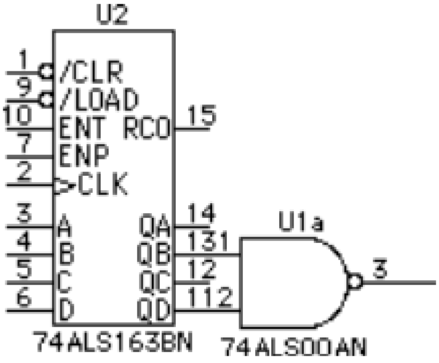
**• Click the mouse once in the part selector text box in the Parts Palette.**

You will notice a text cursor starts to flash in this box.

**• Type the characters "163" on the keyboard.**

**• Double-click on the part "74ALS163(BN,BD)A" in the list.**

## Automatic Pin Connection

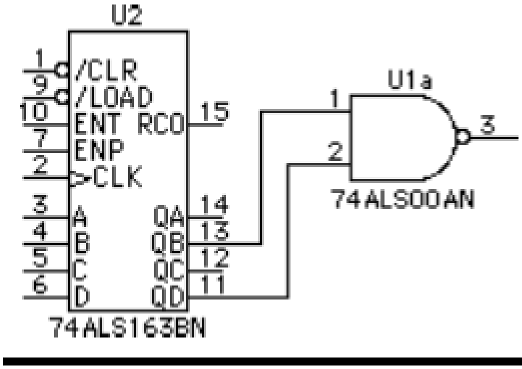


**• Move the cursor so that you can place the 74ALS163 device exactly as shown so that the QB and QD pins just touch the two inputs on the NAND gate.**

**• Click the mouse button to place the device at this point. (Depending on the size of your screen, you may need to use the scroll bar at the bottom of the schematic window to make room on the left.) You will notice the two pins flash to indicate a connection.**

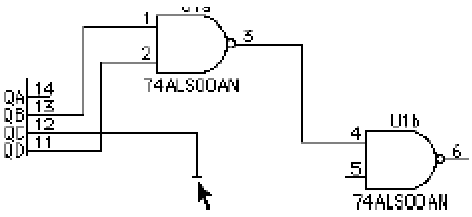
**• Press the spacebar to return to the normal pointer.**

## Automatic Pin Connection (cont'd)



**Click and drag the 74ALS163 device to the left and notice that right-angle lines maintain the connections between the two devices.**

This auto-connection feature can be a convenient way of making connections between large devices, such as 8-bit registers. Just place them so the two sets of pins touch, then drag them apart and all the connections are made automatically.



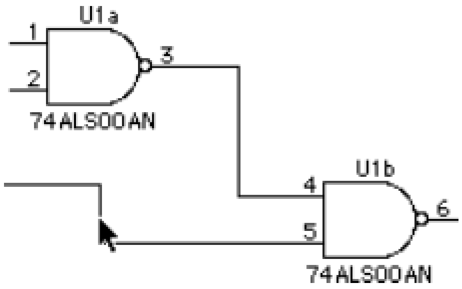
## Wire Editing

**• Using the pointer tool, click and hold on the end of output pin QC on the 74ALS163.**

**• Move to the location shown and release the mouse button. (You may have to modify this procedure slightly depending on the exact positions of your symbols.)**

Notice that a small perpendicular mark is placed at the end of the signal line. All unconnected line ends are marked this way automatically to simplify checking for missed connections.

**• Click and hold on the end of the line just completed and connect it to the lower pin on the NAND gate.**



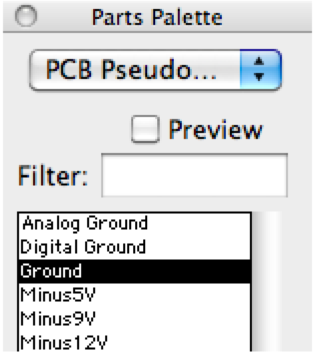
## Wire Editing (cont'd)

NOTE: If you're using a small screen, you can hide the Parts palette at any time by clicking in the "go away" box at its top left corner. To re-display it, use the "Show Part Palette" (Command-L) command in the Libraries sub-menu of the File menu.

**• Using the pointer tool, click and hold at a point midway along the vertical line (or any line) just created. Notice that you can drag this line segment sideways.**

With the pointer tool, clicking at the end of a pin or line segment or at an intersection allows you to extend the signal. Clicking in the middle of a segment allows you to move that segment. The signal drawing tool (+) can be used to draw from any point.

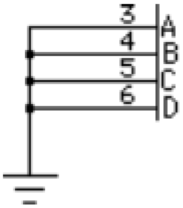
## Power and Ground Connections



- **If necessary, use the scroll bar at the bottom of the schematic window to expose some drawing area to the left of the 74ALS163 device. (If the device is too close to the left edge of the sheet, select the Custom Sheet Info command in the Drawing menu, and enable the "Auto Expand" option.)**
- **Go to the library selection pop-up menu and select the "PCB Pseudo Devices" library.**

The term "pseudo device" is used in DesignWorks to refer to symbols that are edited like devices on the schematic, but are actually symbols used to modify signal connections. Examples of pseudo-devices are power and ground symbols, page connectors and bus breakouts. These items will be discussed in more detail later.

## Power and Ground Connections (cont'd)



- **Select the Ground item in the parts palette, then place one as shown.**
- **Click on the signal tool (+) in the tool palette and wire it to the 74ALS163 device as shown.**

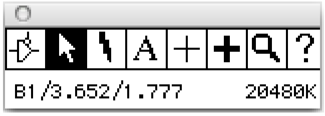
The Ground symbol automatically names the attached net "Ground" and causes it to be logically connected to all other ground nets in the circuit.

## Connector Devices

**• Select the library Connectors and place a DB9F connector symbol below the 74ALS163, as shown. (Depending on the size of your screen, you may need to scroll the schematic window downwards.)**

Connectors can be treated as a single unit, as in this case, or broken up into multiple symbols each with 1 or more connector pins on them. In the netlist, these will be treated as a single device.

## Connecting Signals by Name

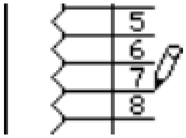


B1/3.652/1.777 20480K

**• Click on the text (A) tool in the tool palette.**

This tool is used to name devices and signals, edit device pin numbers, edit attribute text or create miscellaneous text notations, depending on where it is clicked. The cursor will initially take on a pencil shape (✎), allowing you to point accurately at the item to be named.

## Connecting Signals by Name (cont'd)

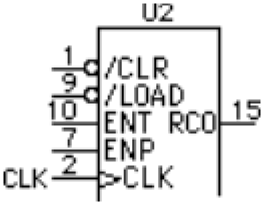


**• Click the pencil tool exactly at the end of connector pin 7, as shown.**

When you release the button a default name is displayed. Unless auto-naming is disabled, every signal is assigned a unique name as it is created. This name is normally only displayed on the diagram when explicitly requested.

**• Type the name "CLK" on the keyboard and hit the Enter key to terminate text entry.**

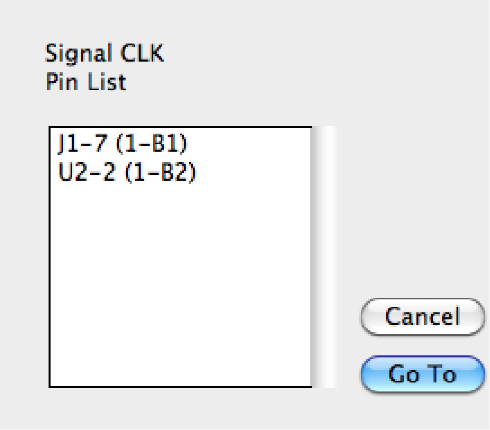
## Connecting Signals by Name (cont'd)



**• Move to the CLK input on the 74ALS163 device and repeat the same procedure, naming this pin "CLK" as well.**

Notice that when you hit the Enter key this time, the signal flashes to indicate a connection has been made with the other CLK label.

## Confirming Connections


Signal CLK
Pin List

J1-7 (1-B1)
U2-2 (1-B2)

**• Click once in the menu bar to return to Point mode.**

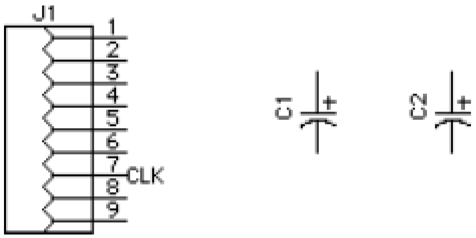
**• Click once on the name CLK that was just typed so that the name itself and the device pin is highlighted.**

**• Select the Get Info command in the Options menu. Click on the Pin List button.**

(There is a shortcut for this procedure that will be mentioned later.) The Pin List box shows you all device pins that are linked by this signal, along with their page number and grid reference.

**• Double-click on any item in this list to display it on the screen.**

The Pin List box will locate items on any page in the circuit, whether connected by signal line, by name, by off-page connector or by bus.



## Discrete Components

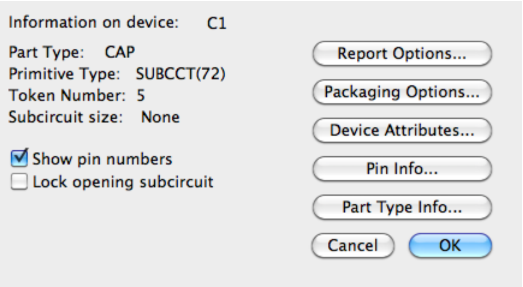
**• Using the library selection pop-up menu, select the library Discretes. Double-click on the item POLCAP and move the cursor into the schematic area.**

**• Press the arrow keys on the keyboard (or, if you don't have arrow keys, click on the item in the tool palette and select a new orientation) to orient the symbol vertically.**

**• Place two capacitors as shown.**

**• Return to the pointer cursor.**

Devices can be rotated to one of 8 orientations (the 4 compass points plus mirrored versions of each). Device text notations can optionally be rotated to match the device.


Information on device: C1

Part Type: CAP
Primitive Type: SUBCCT(72)
Token Number: 5
Subcircuit size: None

☑ Show pin numbers
☐ Lock opening subcircuit

## Setting Component Value

**• Click once on one of the capacitors just placed, so that it is highlighted.**

**• Choose the Get Info command in the Options menu, then click the Device Attributes button.**

The box that appears allows you to view and edit text "attributes" of a device. The list at the left shows the available field names. Clicking on one of these will display the associated value in the text box.

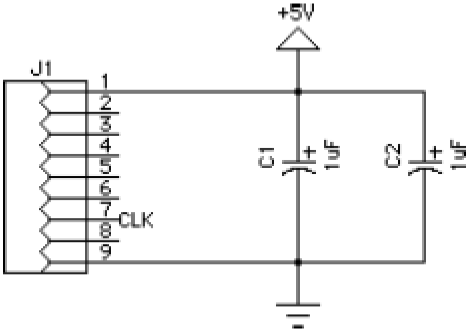
## Setting Component Value (cont'd)

**• Click on the item Value in the field list at left. Type the value "1uF" on the keyboard. Make sure that the Visible box is checked, then click Done.**

**• Repeat this procedure to assign the same value to the other capacitor.**

The component value just entered will now appear adjacent to the device. It can be moved around independently, if desired. We will see in a later tutorial how to edit, rotate, hide and set text style for this text.

## More on Power and Ground

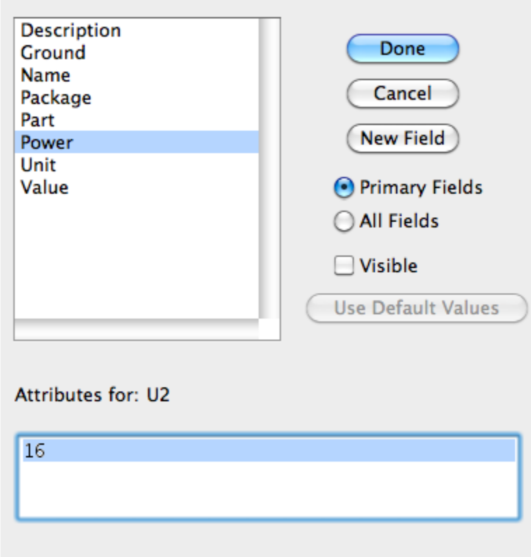


**• Go back to the Pseudo Devs library and place a Ground and a Plus5V symbol. (NOTE: You may have to use the arrow keys again to return to normal orientation.)**

**• Wire them to the capacitors and connector, as shown.**

The Ground and Plus5V symbols are a special class of pseudo-device known in DesignWorks as a "signal connector". They cause all like-named nets to be connected together, even across multiple pages. You can customize your own signal connectors for other types of common connections.

## More on Power and Ground (cont'd)



● Primary Fields
○ All Fields
☐ Visible

Attributes for: U2

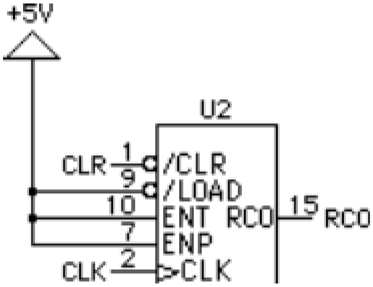
16

- **Click once on the 74ALS163 device so that it is highlighted.**
- **Select the Get Info command in the Options menu, then click on the Device Attributes button.**
- **Select the field Power in the attribute field list and note the field value.**
- **Select the field "Ground".**

You will notice that the fields named "Power" and "Ground" contain the numbers of the power supply pins for this device. You can add other pins to this list, if needed, separated by commas. This allows you to create power and ground connections without showing them explicitly on the diagram. The standard power and ground connections are included in all integrated circuit parts in the DesignWorks libraries.

- **Click the Cancel button on the attributes box, then the Cancel button on the Device Info box.**

## More on Power and Ground (cont'd)



**To complete the counter wiring:**

- **Add a Plus5V symbol and wire it as shown.**
- **Apply names to the CLR and RCO counter pins, as shown.**

## More on Power and Ground (cont'd)

**To complete the connector wiring:**

- **Using the text (A)tool, apply names CLR and RCO to the connector pins shown.**
- **Using the (+) tool, draw a wire from the unconnected NAND gate output to connector pin 6.**

## Generating a Netlist

We have now completed our schematic diagram and the last step is to generate a netlist for PCB layout purposes.

- **Select the Generate Netlist command in the Generic PCB menu.**
- **Using the standard file save box that appears, select a suitable location for the output file.**

You have now saved a text report file that contains a connection list (netlist) for your circuit. You can use any text editor (such as SimpleText) or a word processor to view the file. Note that the assignment of default signal names depends on order of placement and may not exactly match the sample shown at left.

Chapter 14—Report Generation and Scripting on page 363 describes more features of the report generation tool.

```
CLK     J1-7 U2-2
CLR     J1-3 U2-1
Ground   C1-2 C2-2 J1-9 U1-7 U2-3 U2-4
 U2-5 U2-6 U2-8
Plus5V   C1-1 C2-1 J1-1 U1-14 U2-7 U2-9
 U2-10 U2-16
RCO     J1-5 U2-15
SIG1    U1-1 U2-14
SIG2    U1-2 U2-13
SIG3    U1-3 U1-4
SIG6    J1-6 U1-6
SIG20    U1-5 U2-11
```

This completes the tutorial section "The 5-Minute Schematic Diagram".

If you are using a demonstration version of DesignWorks, you will not be able to save the completed circuit, but a similar sample file is provided on the disk.

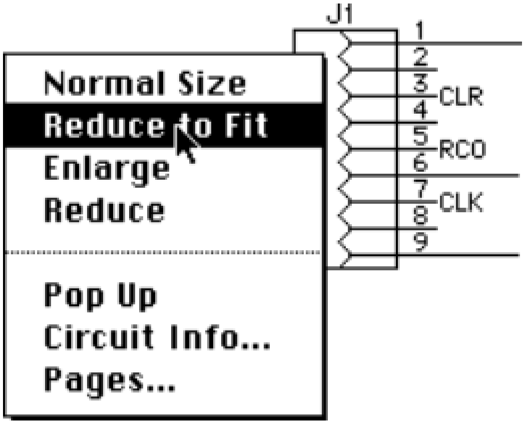
If you are using the full DesignWorks package, then you may wish to use the Save As command in the File menu to save the completed example at this point. This file will be referred to in later sections.

# Advanced Schematic Editing

In this tutorial section, we will cover advanced schematic editing techniques, including:

More editing techniques and shortcuts
Using pop-up menus
Busses and breakouts
Multipage schematics
Sheet size settings
Text notations and graphics
Creating attribute fields
Using the Browser tool to edit attributes
Device packaging.

This tutorial will continue using the file created in the last section. Open the file you created, or use the file "5-Minute Schematic" provided in the "Demo Designs" folder.



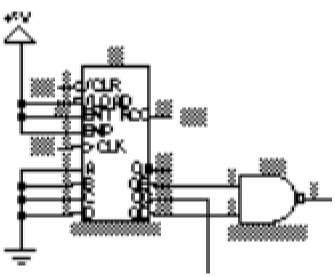
### Navigating Around the Schematic

**• Hold the control key pressed while clicking and holding in an empty area of the schematic (i.e. not on a device or signal line).**

**• In the pop-up menu that appears, select the command Reduce to Fit.**

The screen display will be zoomed out to fit the entire schematic in the window. Holding the control key while clicking in the schematic displays a pop-up menu containing short-cut editing commands. Control-clicking on a device, signal, pin or attribute field will display a special menu for each type of object.

## Navigating Around the Schematic (cont'd)

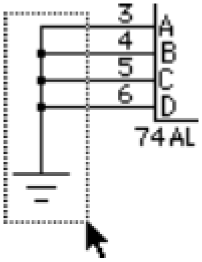


**Click on the magnify ( 🔍 ) tool in the tool palette and then click near the A B C D input pins of the 163 counter.**

Clicking the magnifying glass tool zooms in on that part of the schematic. Some less obvious uses of this tool are:

- Clicking and dragging <u>down and right</u> causes the display to zoom so that the area swept over just fits in the window.
- Clicking and dragging <u>up and to the left</u> a small amount causes the display to zoom out one step.
- Clicking and dragging <u>up and to the left</u> a large amount (more than 1/2 the screen) causes the display to do a Reduce to Fit.

## Deleting a Group of Objects



- **Using the pointer ( ↖ ) tool, click and drag from above and to the left of the ground symbol to below and right of it.**
- **Make sure that the ground symbol itself and the attached signal line are highlighted and no other items.**
- **Hit the Delete key on the keyboard to remove these items.**

## Creating a Bus

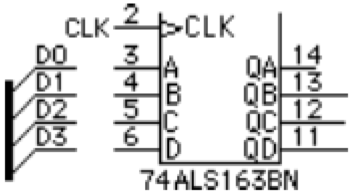
**• Select the New Breakout command in the Options menu.**

**• In the pin list area of this box, type "D3..0".**

**• Click the OK button.**

We used sequentially-numbered signals in this case, but any collection of names can be used, for example "CLK CLR SIZ0 SIZ1". "0..3" places the highest numbered signal at the top, "3..0" places it at the bottom.

## Creating a Bus (cont'd)

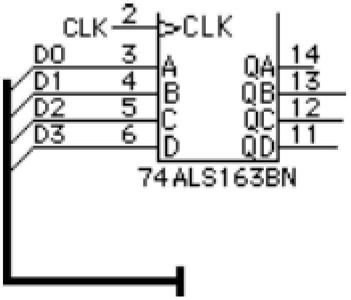


**• Place the "breakout" symbol adjacent to the 163 device, as shown. Use the arrow keys or tool palette if necessary to orient the breakout symbol.**

**• If not done already, connect the 4 breakout pins to the inputs of the 163 device.**

A "breakout" is a pseudo-device symbol that ties any collection of signal lines into a single bus line. You can extend a bus line away from either end of the "spine" of the breakout symbol.

## Creating a Bus (cont'd)



**• Using the pointer (↖) tool, click and hold exactly at the lower end of the "backbone" of the breakout symbol.**

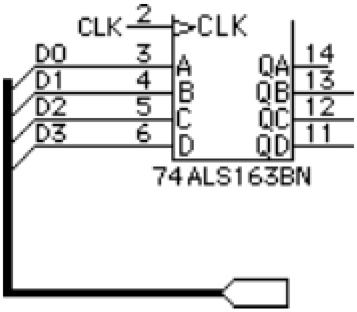
**• Drag down and right from this point, then release the mouse button. This will have created a bus line as shown.**

**• Holding the control key pressed, click and hold on the bus line. In the pop-up menu, select the Signal Info command.**

When the Signal Info command is selected for a bus, a list of the internal signals is displayed.

**• Click OK to close the bus info box.**

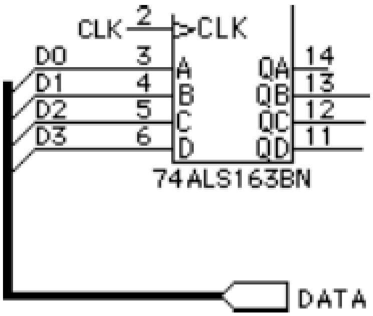
## Creating a Bus (cont'd)



**• In the Parts palette, select a "Page Conn (Bus)" device in the Pseudo Devs library.**

**• Place this symbol as shown so that it connects to the bus that was created above.**

The Page Connector symbol indicates that this bus will be connected to other busses with the same name on other pages of the schematic. Without this symbol, connections by name occur only within a single page.



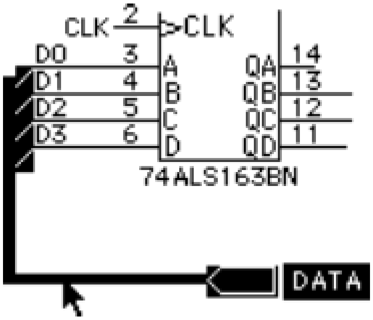
## Creating a Bus (cont'd)

**• Select the text (A) tool in the tool palette.**

**• Click on the Page Connector symbol.**

**• Type the name "DATA" on the keyboard, then hit the Enter key.**

We have now named the bus "DATA". This has no effect on the names of the internal signals, which still retain the names that were assigned in the New Breakout box.



## Using the Clipboard

**• Return to the pointer (↖) tool and click in an unused area of the schematic to ensure that no items are selected.**

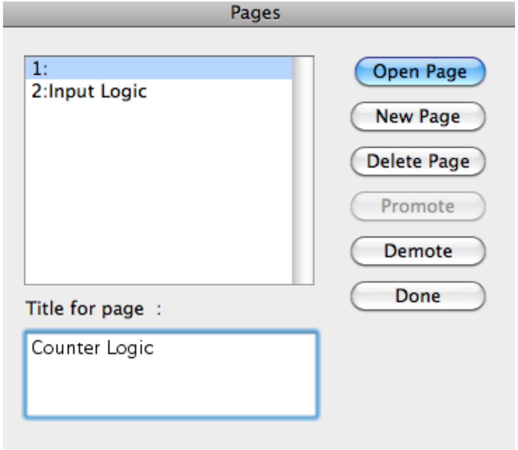
**• Click on the breakout symbol (i.e. on the diagonal line area in the middle of the symbol), to select it.**

**• Holding the Shift key on the keyboard pressed, click on the bus line and then on the Page Connector symbol. All three of these items should now be highlighted.**

Using the Shift key, you can select any collection of items on one page. These items can then be operated on in subsequent menu commands.

**• Select the Copy command in the Edit menu.**

## Adding a Page



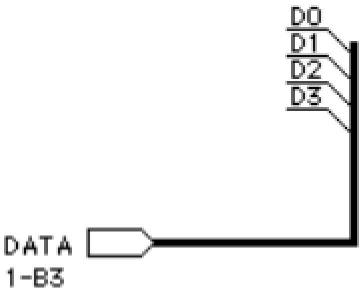
1:
2:Input Logic

Title for page : Counter Logic

- **Select the Pages command in the Drawing menu.**
- **Click on the New Page button.**
- **Enter the title "Input Logic" on the keyboard.**
- **Click on the item "1" in the page list and enter the title "Counter Logic" on the keyboard.**
- **Click on the Done button.**

We have now added a second page to the circuit, which can be edited in its own window. The second page is still logically part of the same circuit and will be stored in the same file. A single circuit can have up to 1000 pages.

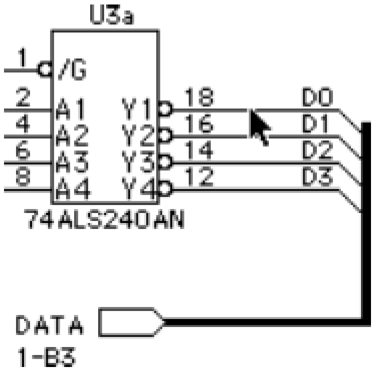
## Adding a Page (cont'd)



- **Select the Paste command in the Edit menu. A flickering image of the bus lines copied from the first page will now appear.**
- **Using the arrow keys, orient this entire group so that the Page Connector is on the left and the breakout is on the right.**
- **Click the mouse button to place the circuit segment in the middle of the page.**

You will notice that a new notation appears next to the Page Connector symbol. This is an automatic page reference which indicates the page and grid position of other page connectors attached by name to this one.

### Adding a Page (cont'd)

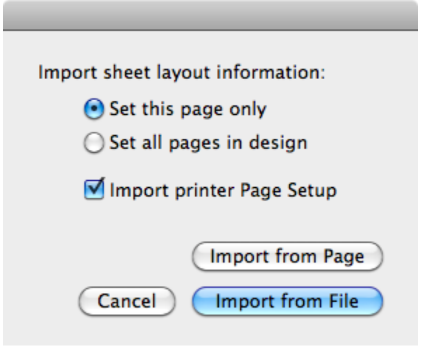


- **Select a 74ALS240 device from the 74ALS {Comp} library and place it so that its outputs connect with the breakout. (NOTE: You may have to use the arrow keys to set the device orientation before placing it.)**
- **Drag this new device away from the breakout to expose some signal lines between the two.**
- **Hold the control key on the keyboard and click and hold on one of these lines.**
- **In the pop-up menu, select the Pin List command.**

The Pin List box will now indicate that the pin on the 240 device is connected via the bus and page connector to the 163 device on Page 1.

- **Double-click on a pin on U2 to return to the first page.**

### Setting Sheet Size


Import sheet layout information:

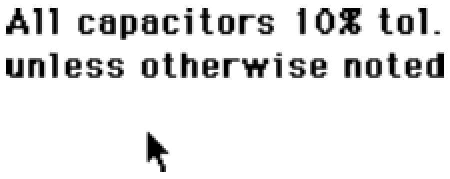
● Set this page only

○ Set all pages in design

☑ Import printer Page Setup

- **Select the Import Sheet Info command in the Drawing menu.**
- **Click on the All Pages in Design button.**
- **Click on the Import from File button.**
- **Locate and open the file ANSI B-Size Template in the Sheet Templates folder.**

You will now see that both pages in our design will be updated with new sheet size settings and border graphics. A "template" file is really just a normal design file that has custom sheet size settings and border graphics placed in it. You can import sheet info from any design file.

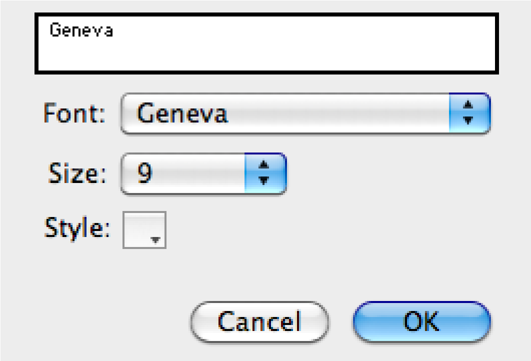


All capacitors 10% tol.
unless otherwise noted

## Adding Text Notations

- **Move to an unused corner of the schematic page.**
- **Select the text (A) tool in the tool palette.**
- **Click in an open area of the schematic (i.e. not on an existing device, signal or text item). You will see a blinking text insertion point appear.**
- **Type "All capacitors 10% tol. unless otherwise noted", or any other notation that suits you. Carriage returns can be used in text notations, if desired.**
- **Type the Enter key to terminate text entry.**

Text notations created in this fashion are stored with the circuit, but are not associated with any device or signal and will never appear in a netlist.



Geneva

Font: Geneva

Size: 9

Style:

## Adding Text Notations (cont'd)

- **Return to the pointer cursor.**
- **Hold the control key and click on the text item that was just created.**
- **Select the Draw Frame option.**
- **Click on the Text Specs button.**
- **Use the controls in this box to set any desired text style.**
- **Click OK on both boxes to return to the schematic.**

Text style can be set individually for random text notations such as this one. Other forms of text such as attribute and pin numbers have global text style settings, but cannot be set individually. See the Design Preferences command in the Drawing menu.

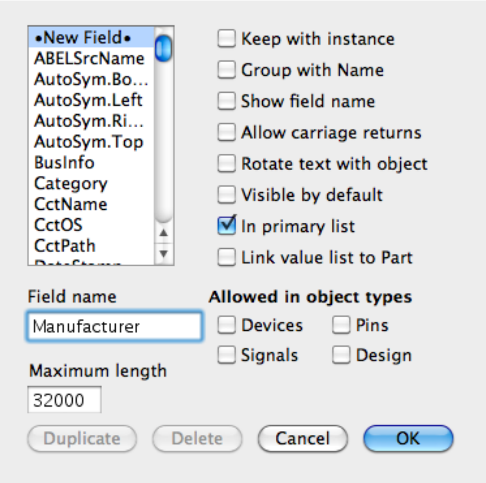
## Adding Graphics

**• Go to your Scrapbook (in the Apple menu) or to any other graphics program and Copy a picture item.**

**• Return to DesignWorks and select the Paste command in the Edit menu. Place the graphic item in any desired location.**

Graphic objects can be pasted from any other standard Macintosh application. They can be used for your company logo, for mechanical drawings, or for any other graphic notations. This type of graphical object is for notation only and will not appear in a netlist. The DevEditor tool is used to create device symbols.

## Creating Attribute Fields



☐ Keep with instance
☐ Group with Name
☐ Show field name
☐ Allow carriage returns
☐ Rotate text with object
☐ Visible by default
☑ In primary list
☐ Link value list to Part

Field name: Manufacturer

Maximum length: 32000

**Allowed in object types**
☐ Devices ☐ Pins
☐ Signals ☐ Design

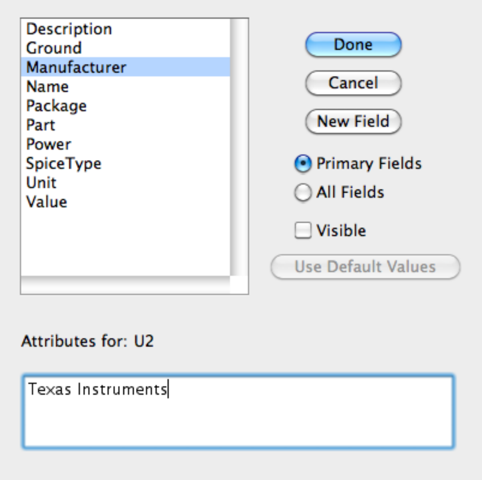
**• Select the Define Attribute Fields command in the Options menu.**

**• Type the name "Manufacturer" in the Field Name box, but <u>don't</u> hit Return or Enter yet.**

**• Click on the Devices option under "Allowed in object types".**

**• Click on the OK button.**

You have now defined a new attribute field that will be associated with devices. A separate value can be entered for each device in your design. The other options in this box determine how the value is stored and displayed on the diagram.



● Primary Fields
○ All Fields
☐ Visible

Attributes for: U2

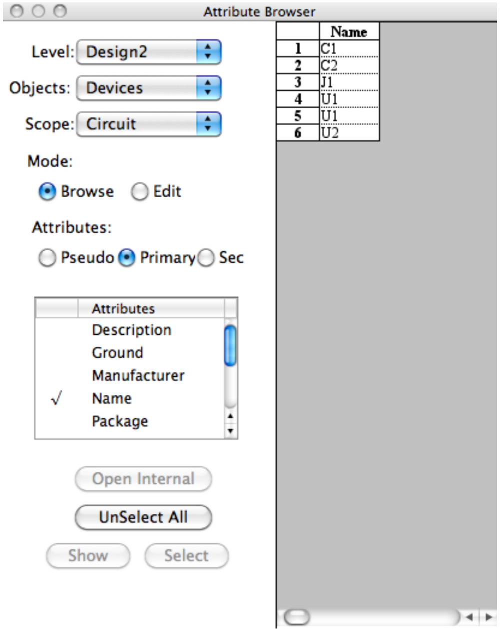
Texas Instruments

## Creating Attribute Fields (cont'd)

- **Hold the control key on the keyboard and click <u>and hold</u> on any device in the schematic.**
- **In the pop-up menu, select the Attributes command.**
- **Type a value for the Manufacturer field in the text box.**
- **Click on the Done button or hit the Enter key.**

If desired, you can enter values for this field into all the devices in the schematic. The Browser tool is a quicker method of doing this and is described in a later tutorial.

Attribute values can be entered for devices, signals, pins and for the design itself. These values can be displayed on the schematic and extracted in netlists and bills of materials for external use.



Level: Design2
Objects: Devices
Scope: Circuit

Mode:
● Browse ○ Edit

Attributes:
○ Pseudo ● Primary ○ Sec

√ Name

| | Name |
|---|---|
| 1 | C1 |
| 2 | C2 |
| 3 | J1 |
| 4 | U1 |
| 5 | U1 |
| 6 | U2 |

## Using the Browser Tool

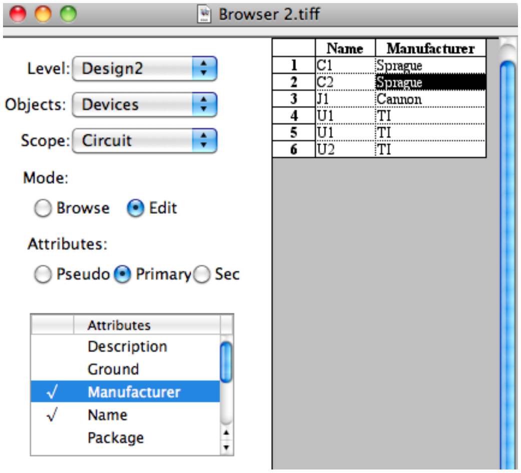
- **Click in an unused area of the schematic to ensure that no devices or signals are selected.**
- **Select the Browser item in the Tools menu.**

The Browser displays the selected type of object in a spreadsheet format. This allows you to easily locate items and edit attributes without having to search for them on the schematic sheet.

- **Click on any device name in the list and click the Show button.**

Notice that the schematic sheet is scrolled to display the selected device. The Browser can also view Signals or Pins (see the Object Type pop-up menu).

## Using the Browser Tool (cont'd)



Level: Design2

Objects: Devices

Scope: Circuit

Mode:
○ Browse ● Edit

Attributes:
○ Pseudo ● Primary ○ Sec

| | Attributes |
|---|---|
| | Description |
| | Ground |
| √ | Manufacturer |
| √ | Name |
| | Package |

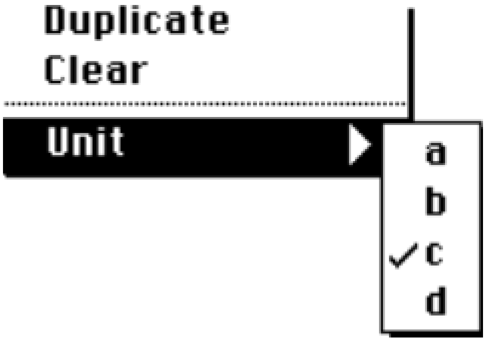
| | Name | Manufacturer |
|---|---|---|
| 1 | C1 | Sprague |
| 2 | C2 | Sprague |
| 3 | J1 | Cannon |
| 4 | U1 | TI |
| 5 | U1 | TI |
| 6 | U2 | TI |

- **Click on the Primary pop-up menu and select the Manufacturer field, or any other desired field.**
- **Click on the Mode pop-up menu and select the Edit item. This places the Browser in Edit mode, allowing you to change values in the table.**
- **Click on any boxes in the "Manufacturer" column and entered any desired values.**
- **Use the Return key to move down to other cells in the same column.**

Changes made to data in the Browser cause the Schematic to be updated immediately. These changes cannot be canceled or undone!

- **Close the Browser by clicking in the "go-away" box at the top left corner of the window.**

## Device Packaging



- **Hold the control key on the keyboard and click and hold on one of the NAND gate devices on page 1.**
- **Move down to the Unit sub-menu, near the bottom.**
- **Select Unit "c" in the Unit sub-menu.**

The Unit sub-menu contains a list of all gate units available for this part type. Selecting a different unit will update the pin numbers to correspond.

NOTE: If you select a Unit that is already in use, the program will allow the change, but Auto-Packaging will be disabled. You can re-enable it using the menu items in the Packaging sub-menu of the Options menu.

### Device Packaging (cont'd)

**• Command-click on the same device as in the previous section and select the Device Info command.**

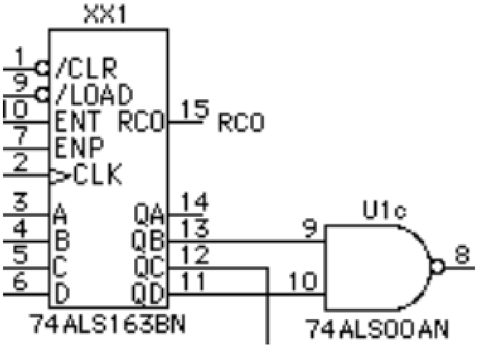
**• Click on the Packaging Options button.**

**• Select the "Lock and check package and unit" option.**

**• Click OK on both boxes.**

The Packager has now been instructed to keep this device name unchanged but check it for conflicts in future packaging operations.

### Device Packaging (cont'd)



**• Select the Device Naming and Packaging Options command in the Naming and Packaging sub-menu of the Options menu.**

**• Type the Default name prefix "XX" and click the OK button.**

**• Select the Repackage Design command in the Naming and Packaging sub-menu of the Options menu.**

**• Click the OK button in the confirmation box.**

Notice that new names have now been assigned to all devices except the one that we marked as "Lock and Check" (and anything else in the same package). Names are assigned starting in the "A1" grid of the schematic, working toward the opposite corner.

This concludes the tutorial section "Advanced Schematic Editing". The remaining tutorials deal with specific areas of the program and do not make further reference to this file.

# Device Symbol Editing and Hierarchical Design

In this tutorial section we will cover some of the DesignWorks features for creating and edting device symbols and associating internal circuits with them.

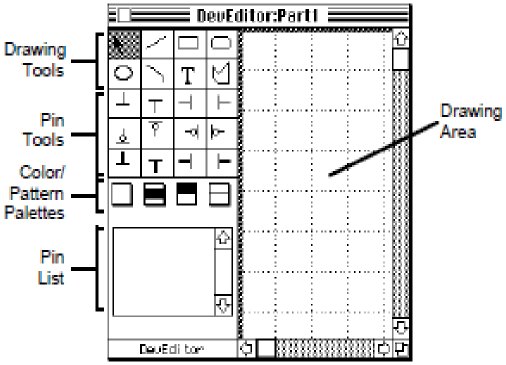
## Creating a New Library

**• Command-click on the Parts palette and select the New Lib command in the pop-up menu.**

**• Create a new library called My Lib in the DesignWorks 4.x folder.**

Device library files hold collections of part symbols along with associated pin function information, default attribute values and internal circuit definitions. A single library can contain from one to thousands of part definitions, to suit your needs.

## Creating a Device Symbol



**• Select the DevEditor item in the Tools menu.**

The DevEditor window contains a drawing area for your symbol, a tool palette and a pin list. The tool palette includes standard Macintosh drawing tools plus special items for normal, inverted and bus pin placement.

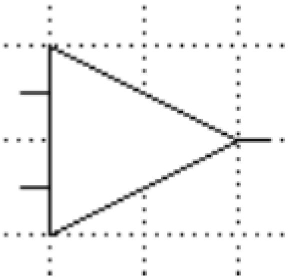
## Creating a Device Symbol (cont'd)

**• Click on the polygon ( ) tool in the tool palette.**

**• Draw a symbol similar to the one shown by clicking once at each corner point and then double-clicking to terminate the polygon.**

The position of the symbol in this window is not important.

## Creating a Device Symbol (cont'd)



**• Select the ( ┤ ) pin tool.**

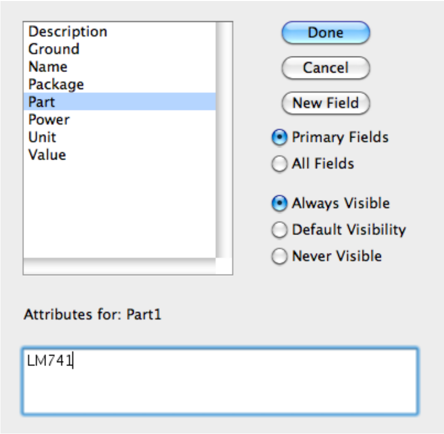
**• Place input pins on the symbol by clicking at the positions shown.**

**• Select the ( ├ ) pin tool.**

**• Place an output pin as shown.**

Note: The crossbar portion of the T pin tool only appears during placement and dragging for alignment purposes.

## Setting Default Part Attributes



● Primary Fields
○ All Fields

● Always Visible
○ Default Visibility
○ Never Visible

Attributes for: Part1

LM741

**• Select the Part Attributes command in the DevEdit menu.**

**• Select the Part field in the list.**

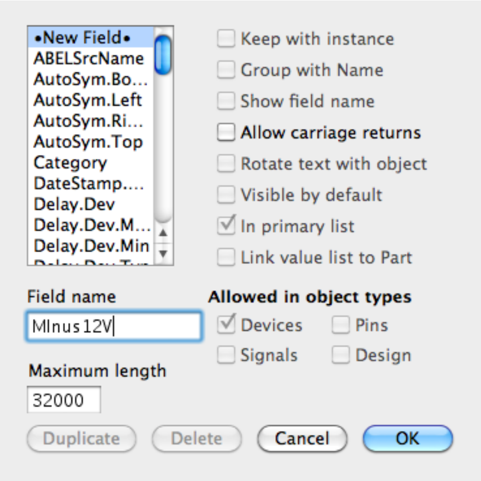
**• Enter the value "LM741" or any other value.**

**• Select the Package field in the list.**

**• Enter the value "NAT8DPN" or other package code.**

These attribute values will appear as the defaults when this part is used on a schematic. If desired, these values can be overridden for each individual device.

## Setting Default Part Attributes (cont'd)



☐ Keep with instance
☐ Group with Name
☐ Show field name
☐ Allow carriage returns
☐ Rotate text with object
☐ Visible by default
☑ In primary list
☐ Link value list to Part

Field name
MInus12V

Allowed in object types
☑ Devices ☐ Pins
☐ Signals ☐ Design

Maximum length
32000

**• Click on the New Field button in the attributes box.**

**• Type the field name "Plus12V" then type the Tab key to terminate this value.**

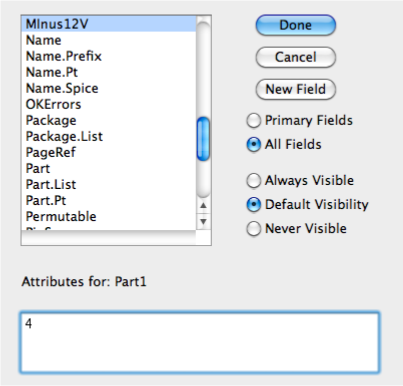
**• Select the "•New Field•" item at the top of the field list again.**

**• Enter the field name "Minus12V".**

**• Click the OK button.**

If you create multiple devices in succession, you only need to perform the "New Field" operation once since the DevEditor keeps the same field list for all devices.

## Setting Default Part Attributes (cont'd)



○ Primary Fields
● All Fields

○ Always Visible
● Default Visibility
○ Never Visible

Attributes for: Part1

4

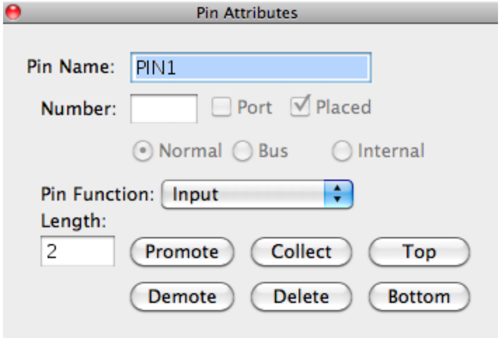
**• Enter the value "4" for the new Minus12V field.**

**• Enter the value "7" for the "Plus12V" field.**

These entries will determine the default power connections for this device in a netlist.

**• Click on the Done button to close the attributes box.**

## Entering Pin Names and Numbers


Pin Attributes

Pin Name: PIN1

Number: ☐ Port ☑ Placed

● Normal ○ Bus ○ Internal

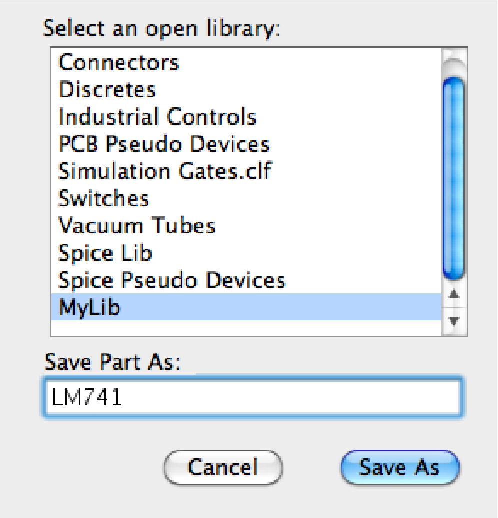
Pin Function: Input

Length: 2

- **Double-click on the PIN1 item in the pin list.**
- **In the Pin Info palette that appears, enter the pin name "INA" and pin number "2".**
- **Hit the Enter key on the keyboard to move to the next pin in the list.**
- **Enter name "INB" and pin number "3".**
- **Hit Enter again to see the last pin and enter pin name "OUT" and number "6".**
- **Hit Enter to complete the last pin, then close the Pin Info box by clicking on the "go-away" box in its top left corner.**

We have now entered default values for the pin numbers that will appear in a netlist. These can be edited on the schematic for individual pins, if desired.

## Saving and Using the Part


Select an open library:

Connectors
Discretes
Industrial Controls
PCB Pseudo Devices
Simulation Gates.clf
Switches
Vacuum Tubes
Spice Lib
Spice Pseudo Devices
MyLib

Save Part As: LM741

- **Select the Save Part As command in the File menu.**
- **Enter the part name "LM741" or any other desired name.**
- **Double click on the "My Lib" library in the list to select a destination.**
- Close the DevEditor by clicking the "go-away" box at the top left corner of the window.

## Saving and Using the Part (cont'd)

**• If it is not already selected, select the My Lib library in the pop-up library list in the Parts palette.**

**• Double-click on the newly-created part and place one in the schematic.**

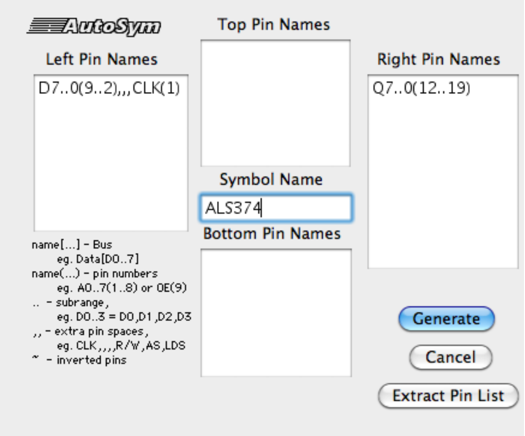
NOTE: A warning box will be displayed indicating that new attribute fields are being defined.

**• Select the Attribute Probe ( ? ) tool in the schematic tool palette.**

**• Click on the new device to verify that the default attribute values appear.**

## Auto-Creating a Symbol

For standard types of rectangular symbols, the Auto Create feature will generate a symbol for you in seconds.



Left Pin Names: D7..0(9..2),,,CLK(1)

Right Pin Names: Q7..0(12..19)

Symbol Name: ALS374

Bottom Pin Names:

name[...] - Bus
eg. Data[D0..7]
name(...) - pin numbers
eg. A0..7(1..8) or OE(9)
.. - subrange,
eg. D0..3 = D0,D1 ,D2,D3
,, - extra pin spaces,
eg. CLK,,,,R/W,AS,LDS
~ - inverted pins

**• Select the DevEditor command in the Tools menu.**

**• Select the Auto Create Symbol command in the DevEdit menu.**

**• In the "Left Pins" box, type the text "D7..0(9..2),,,CLK(1)".**

"D7..0" will generate a set of 8 pins named D7, D6, etc. "(9..2)" are the corresponding pin numbers. The three commas indicate that we want extra space between these pins. "CLK(1)" creates a single pin called CLK with pin number 1. The pin numbers can be omitted, if desired.

**• In the "Right Pins" box, type the text "Q7..0(12..19)".**

**• In the Name box, type "ALS374", or any other desired symbol name.**

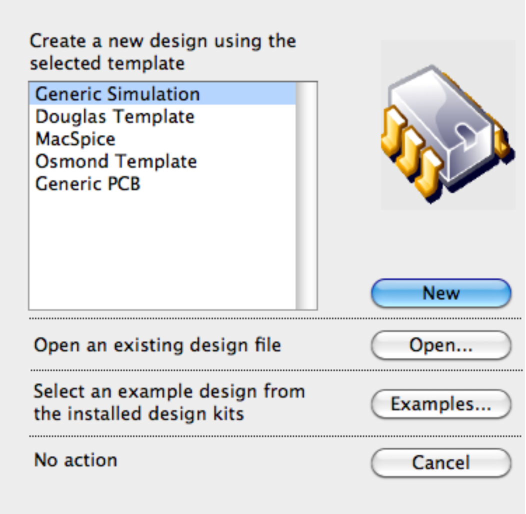
**• Click the Generate button.**

## Auto-Creating a Symbol (cont'd)

The auto-generated symbol should now display the pins and pin numbers entered above.  These items can be edited using the drawing tools and Pin Info box, if desired.

- **Select the Save Part As item in the file menu and save the new part to the My Lib library.**
- **Close the DevEditor window.**

## Creating a Hierarchical Block


Create a new design using the selected template
Generic Simulation
Douglas Template
MacSpice
Osmond Template
Generic PCB
New
Open an existing design file
Open...
Select an example design from the installed design kits
Examples...
No action
Cancel

- **Select the New Design command in the File menu.**
- **Select the Generic Simulation template.**

NOTE: If this template does not appear in the list, the Generic SPICE design kit is not activated.  See “Choosing and Activating a Design Kit” on page 337.

- **Click the New button.**

This template enables hierarchy, which will allow us to create nested circuit blocks inside device symbols.

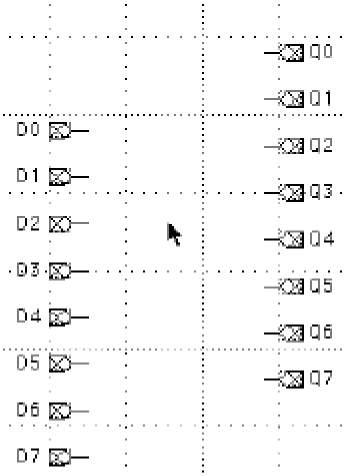
## Creating a Hierarchical Block (cont'd)

**• In the Parts palette, double-click on the ALS374 device that was just created.**

**• Place two of these devices on the schematic, as shown.**

When new symbols are made in the DevEditor, they are set by default to be "Sub-Circuit" devices, meaning that an internal circuit can be created in them, even if none is provided initially.

## Creating a Hierarchical Block (cont'd)



**• Double-click on either of the ALS374 devices that were just placed.**

**• Click OK in the warning box to create an internal circuit.**

A new window will now appear showing the internal circuit for this block. When a new internal circuit is created, "port connector" symbols are automatically laid out to match the pins on the parent symbol. Any signal connection made to a port connector becomes logically connected to the pin with the same name on the parent symbol.

### Creating a Hierarchical Block (cont'd)

**• Close the internal circuit window by clicking on its "go-away" box.**

**• A box will appear allowing you to select the appropriate action in updating the part definition. Click on the Update button.**

The "Update" option will cause the definition of all devices of the same type to be updated throughout the design. Double-clicking on any other device of the same type will now reveal the same internal circuit.

**IMPORTANT:** Unless you are an experienced user of hierarchical design concepts, please refer to “Hierarchical Design” on page 121 before using these features.

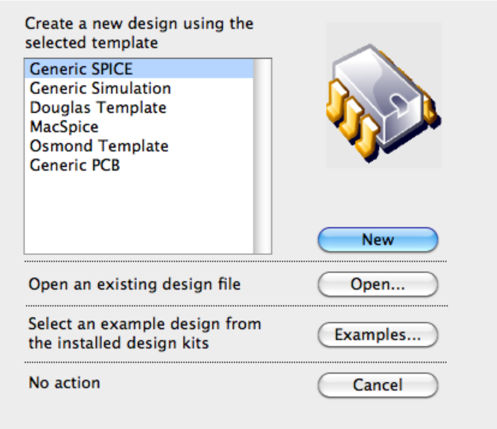
This concludes the tutorial section on Device Symbol Editing and Hierarchical Design. More information on hierarchical design features is found in the Appendix.

# Using DesignWorks with SPICE-based Simulators

DesignWorks makes an ideal schematic entry tool for SPICE-based simulators and others with similar formats, such as PSpice™, SMASH™ and standard SPICE.

In this tutorial section, we will create a simple SPICE circuit and write out the complete netlist. You can then try this netlist out in your SPICE package.

## Creating a SPICE Design



Create a new design using the selected template

- Generic SPICE
- Generic Simulation
- Douglas Template
- MacSpice
- Osmond Template
- Generic PCB

Open an existing design file

Select an example design from the installed design kits

No action

**• Start the DesignWorks program or, if it's already running, select the New Design command in the File menu.**

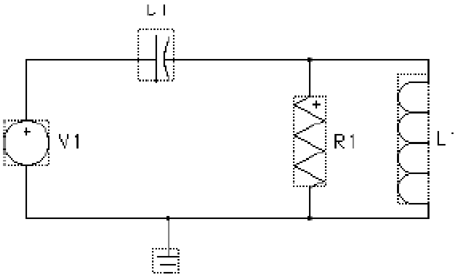
**• Select Generic SPICE template.**

NOTE: If this template does not appear in the list, the Generic SPICE design kit is not activated. See “Choosing and Activating a Design Kit” on page 337.

**• Click on the New button to create an empty design window.**

This template enables an option that auto-names devices using a SPICE prefix supplied in the library. In the SPICE Devices library provided with DesignWorks, this

## Placing SPICE Devices

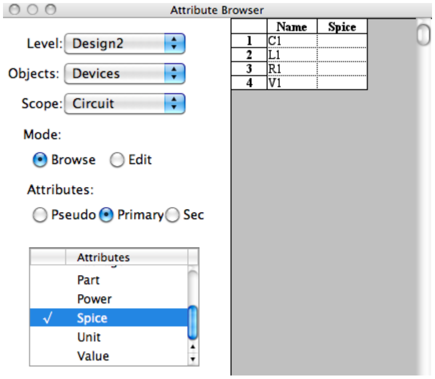


**• Select Spice Lib in the library selection pop-up menu in the Parts palette.**

**• Using the devices Ind Volt. Src, Spice Ground, Spice Res, Spice Cap and Spice Ind, create the circuit shown at left.**

Some of the passive components in this library, such as the Spice resistor, as marked to ensure consistent polarity for current-measuring purposes.

## Entering SPICE Parameters



Level: Design2

Objects: Devices

Scope: Circuit

Mode:

● Browse ○ Edit

Attributes:

○ Pseudo ● Primary ○ Sec

| | Name | Spice |
|---|---|---|
| 1 | C1 | |
| 2 | L1 | |
| 3 | R1 | |
| 4 | V1 | |

**• Select the Browser command in the Tools menu.**

**• Click on the Sort button.**

**• Click on the Mode pop-up menu and switch the mode to Edit.**

**• In the Primary pop-up menu, select the field Spice.**

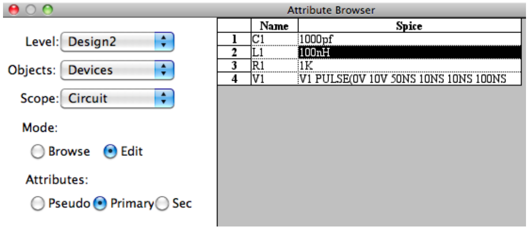
All text entered in the Spice attribute field will be included in the netlist with the associated device.

## Entering SPICE Parameters (cont'd)

**• Click in the box under Spice and adjacent to device C1 in the list.**

**• Enter the value "1000pF".**

NOTE: You can resize the rows and columns by clicking and dragging on the dividing lines in the border area of the spreadsheet.



Level: Design2

Objects: Devices

Scope: Circuit

Mode:

○ Browse ● Edit

Attributes:

○ Pseudo ● Primary ○ Sec

| | Name | Spice |
|---|---|---|
| 1 | C1 | 1000pf |
| 2 | L1 | 100nH |
| 3 | R1 | 1K |
| 4 | V1 | V1 PULSE(0V 10V 50NS 10NS 10NS 100NS |

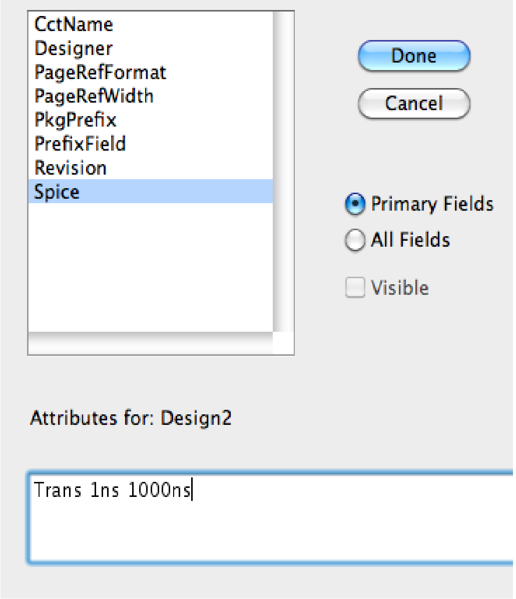
**• Enter values for the remaining devices as follows:**

L1 100nH
R1 1K
V1 PULSE(0V 10V 50NS 10NS 10NS 100NS 1US)

**• Click Return or Tab after entering the last value to be sure the schematic is updated.**

**• Close the Browser by clicking in the window's close box.**

## Entering SPICE Parameters (cont'd)



Attributes for: Design2

Trans 1ns 1000ns

- **Select the command Set Design Attributes in the Options menu.**
- **Select the Spice field in the list.**
- **Enter the text ".TRAN 1ns 1000ns". If you are using PSpice, you may wish to hit the Return key and enter a ".PROBE". line to generate output for the Probe display program.**

Any text entered in the design's Spice field will appear at the beginning of the SPICE netlist. This can be used to enter SPICE commands, model definitions, etc. Multiple lines can be entered if desired using the Return key. You can also use the Command-V (paste) keyboard combination to paste larger numbers of lines into this box.

- **Click on the Done button to close the attributes box.**

## Creating a SPICE Netlist

```
Design2   Thursday, October 22, 1993   11:52 AM
.TRAN 1ns 1000ns
C1 SIG3 SIG7 1000pF
L1 SIG7 SIG8 100nH
R1 SIG7 SIG8 1K
V1 SIG3 SIG8 PULSE(0V 10V 50NS 10NS 10NS
100NS 1US)
.END
```

- **Select the Generate Netlist command in the Generic SPICE menu.**
- **Select an appropriate output file name and click the Save button.**

The netlist file just created can now be fed directly to your simulator.

This concludes the tutorial section Using DesignWorks with SPICE-based Simulators.

DesignWorks can also directly create hierarchical SPICE netlists with .SUBCKT definitions. Hierarchical block symbols can be nested to any desired depth.

# Basic Procedures

# 4

This chapter provides a quick overview of a DesignWorks schematic and how it is stored. It will then provide simple procedures for creating, saving, opening, navigating around and printing documents, and look at the important issue of backup procedures.

DesignWorks deals with a number of different types of files. Far and away the most important type is the design file, which contains all the stored data of your schematics. Operations on design files are discussed in this chapter. More information on the other types of data files that you may encounter can be found in these parts of the manual:

Library Files: Chapter 12—Device Symbols on page 271.

Report Script Files: DesignWorks Script Language Reference (separate manual on disk).

Template Files: Chapter 13—Design Kits and Sheet Templates on page 335.

Setup Files: Appendix D—Setup File Format on page 387.

## Design Structure

In DesignWorks the term "design" refers to a complete, independent logical entity which is saved in a single file. The following rules outline how a design is stored:

A single design is stored in a single file and no logical connections are made between designs. All information required to display and edit a design is stored in the design file.

A design never makes reference to external library files. When a library device is used, all information needed is read from the library and stored with the design. Changing the original library definition <u>will not</u> automatically update the design. More information on the relationship between symbol libraries and designs can be found in Chapter 12—Device Symbols on page 271.

When a design file is opened, the entire contents of the design are read into memory. This means that design sizes are limited by the available memory in your computer and increasing the memory allocated to the program will increase the size of the designs you can work with.

A number of user-settable parameters are stored with the design and affect the entire design when changed. These include:

attribute and pin number text style settings

the attribute definition table

signal and device auto-naming settings

display options, such as crosshairs and printed page breaks

page reference format

hierarchy mode

In a hierarchical design (that is, one in which a symbol can represent a nested circuit block), all hierarchy levels are stored in a single file. More information on hierarchical design is in Chapter 11—Hierarchical Design on page 235.

For the balance of this chapter, we'll assume we're dealing with a simple, "flat" design (i.e. not hierarchical) consisting of a single circuit level.

## What is a Circuit?

A circuit in DesignWorks has these characteristics:

A circuit can be drawn on one or more pages (up to 1000 in this version). You can elect to draw the entire circuit on a single page or divide it up functionally onto a number of pages.

Each page is viewed in a separate circuit window and any or all pages in a circuit can be displayed on the screen simultaneously.

A page is drawn on the screen as if it was a single piece of paper, although it may have to be broken up into a number of individual sheets of paper for printing or plotting.

Logical connections can be made between pages using the Page Connector device allowing for full simulation and netlisting across pages.

## Types of Objects in a Circuit

A DesignWorks circuit is made up of four types of entities: devices, signals, text objects and picture objects.

A device is an object having a symbol, signal connection points called “pins”, and optional text attributes, internal circuit and simulation information. A device in DesignWorks can correspond to a physical device in a circuit, or it can represent a sub-circuit block or a *pseudo-device*, such as a page connector or bus breakout.

A pin is a connection point on a device. A pin is not an independent entity since it only exists as part of a device and cannot be created or deleted separately. However, pins can have attributes, pin numbers and other parameters that may be different from pin to pin on the same device. A *bus pin* is a special type of pin that represents an arbitrary number of internal pins. The internal pins are not visible on the schematic but can still have the same logical properties as other pins.

A signal is a conductive path between devices. Signal connections can be made visually by drawing lines between device pins, or logically by name or bus connection.

A text object is used to place a title block or other notation on the diagram. Text can be typed and edited directly within DesignWorks, or can be created externally and pasted onto the diagram from the clipboard. Text objects are not associated with any other object and are not accessible through net or component lists. The attribute facilities should be used to associate text with specific devices or signals.

A picture object is used to place any graphics item imported from another program or the DevEditor tool via the clipboard. The Schematic tool cannot edit picture objects directly, other than moving, deleting and copying them whole. They can, however, be pasted into a DevEditor window, edited using the device symbol tools, then copied and pasted back into the schematic.

# Creating a New Design

A new design is created by selecting the New Design command in the File menu and choosing from one of the available templates. The new design will consist of an empty circuit with a single page. This page will appear in a circuit window entitled “Design1:1” (the “:1” portion indicates that this is page 1). The New Design command does not create a disk file. The design exists only in memory until you do a Save Design As command.

Your circuit diagram is created by first placing one or more devices in the circuit window (as described below), and then interconnecting the device pins with signal connections.

## Choosing a Template

DesignWorks does not have a fixed set of templates, but rather displays the ones that are available in the active design kits. See the accompanying sidebar for more information on how this is done. A template may contain a variety of initial settings for options and items such as a sheet border and title block, that make it easier for you to get started on a design. We encourage you to create your own templates that suit your application, once you are comfortable with the basic operation of DesignWorks.

***See Chapter 13—Design Kits and Sheet Templates on page 335 for more information on creating design templates.***

In the meantime:

> If you have just installed DesignWorks: When you select the New Design command, you will see a number of generic templates that are provided for tutorial purposes. We do not recommend using these for a major design, but you can choose the one that best suits your application just to try out the package. Before you start any significant design entry, you should refer to Chapter 6—Before Starting a Major Design on page 117.
>
> If you are an experienced user of earlier versions of DesignWorks: You should refer to “Creating a Design Kit” on page 341 for information on setting up templates with your preferred initial settings.
>
> If you are a new user of DesignWorks, but you have walked up to an

already-installed package, the list of templates you see will depend on which design kits are installed. If there is any doubt about the usage of the templates that you see, you should consult the person who installed them, or refer to Chapter 13—Design Kits and Sheet Templates on page 335 for more information on the types of settings that should be in a template.

## Opening a Design

A design file can be opened in one of two ways:

Select the Open Design command in the File menu. This command allows a design to be opened from a disk file using the standard Macintosh file open box.

Locate the file using the Finder and double-click on the file's icon. If DesignWorks is not already running, it will be started up. If it is already running, it will come to the front and the selected file will be opened.

When you open a file, the circuit data is read into memory in its entirety and no more access to the disk file is required. DesignWorks will let you open multiple copies of the same file and makes no attempt to restrict you from writing any of them back to the same file. If you do this, it is up to you to keep track of which windows have been updated and what file you want to save them into.

There is no fixed limit on the number of designs that can be open at once, although the complete contents of all open designs must fit into memory.

### Compatibility With Older Versions

DesignWorks 4.7 can directly read files created by all DesignWorks 3.x versions on Macintosh and Windows. The files created by DesignWorks 4.7 can be read by DesignWorks 3.1.5 on Macintosh or Windows, except that a warning message will be displayed indicating that some data has been skipped. This warning can be ignored as it pertains only to some global settings that are stored in the 4.0 file that are not relevant in 3.x.

## Templates and Design Kits

An important concept in DesignWorks 4.7 is the design kit. A design kit consists of all the items needed to customize DesignWorks for a particular application, including:

Sheet templates
Example designs
Symbol libraries
Netlist and report formats
Error checking scripts
Data entry boxes
Back annotation formats
Custom menus.

A design kit groups everything together under one heading and makes it easy to install or remove a complete design kit in one simple operation.

### Creating a Design Kit

DesignWorks 4.7 will be shipped with a variety of design kits for the various common applications. However, most users will want to customize these to some degree, whether it be simply adding their company logo to the sheet templates, or creating an extensive set of libraries and scripts for an exotic design process.

***Complete information on creating a design kit is given in Chapter 13—Design Kits and Sheet Templates on page 335.***

### How Design Templates Work

A design template is simply a normal design file that has its sheet size, attribute fields, hierarchy mode and other settings pre-defined for the application at hand. The simplest way to create a template may be to take an existing design that is set up the way you like it, delete all the circuit elements and extra pages out of it and save it in the appropriate template directory.

When you select the New Design command and select one of the template files listed, DesignWorks just reads the file in the normal way, then renames it “Design1” (or the next available number), and disassociates it from the original file so that it cannot accidentally be Saved on top of the template. In all other respects, New Design is the same as doing an Open on the template design. All the settings and contents of the design template file become part of the new design.

# Navigating Around a Schematic Page

In addition to the standard scroll bars, DesignWorks has these convenient features for moving around a diagram:

> Auto-Scrolling - Whenever the mouse button is depressed and moves close to the edge of a schematic window, the window automatically scrolls to expose more area on that side.
>
> Auto-Accelerating Scroll Bars - The scroll bars on a Schematic window operate in the same manner as all Macintosh applications with the following additional feature: holding the mouse button depressed in the scroll arrow at either end of a scroll bar will engage the "auto-accelerate" feature. Each scroll step will be larger than the previous one so that scrolling proceeds more and more rapidly in the direction of the arrow.

## Zooming In and Out

### Controlling Zoom by Menu Commands

The Drawing menu contains four menu items for controlling the zoom factor of a window: These commands control screen display only and have no effect on the stored circuit information, printed output, or graphics files. Due to the integer calculations that are done by DesignWorks and by the Macintosh system, device symbols and text may be displayed rather crudely at scale factors other than 100%. It is best to do most editing at normal size to ensure that everything lines up as you would expect.

| | |
|---|---|
| **Normal Size** | Sets the screen scale to 100%. |
| **Reduce to Fit** | Sets the scale factor and centers the display so that the entire diagram fits on the screen. If the size of the diagram and the size of the window is such that this would require a scale factor of less than 20%, then the scale is set to 20% and the diagram is centered. If the diagram fits completely in the window at 100%, then the scale is set to 100% and the diagram is centered. |
| **Enlarge** | Increases the scale factor by about 20%, causing the diagram to appear larger on the screen. If this causes the setting to exceed the maximum of 200%, it is set to the maximum. |

| | |
|---|---|
| **Reduce** | Decreases the scale factor by about 20%, causing the diagram to appear smaller on the screen. If this causes the setting to go below the minimum of 20%, it is set to 20% |

### Controlling Zoom with the Magnifying Glass Tool

The 🔍 item in the tool palette is a powerful tool for moving around in a schematic diagram. It can be used to zoom both in and out and can control the exact area displayed on the screen.

Clicking and releasing the mouse button on a point on the diagram will zoom in to that point by one step.

Clicking and dragging the mouse down and to the right zooms in on the selected area. The point at which you press the mouse button will become the top left corner of the new viewing area. The point at which you release the button will become approximately the lower right corner of the displayed area. The circuit position and scaling will be adjusted to display the indicated area.

Clicking and dragging the mouse upward and to the left zooms out to view more of the schematic in the window. The degree of change in the scale factor is determined by how far the mouse is moved. Moving a small distance zooms out by one step (equivalent to using the Reduce command). Moving most of the way across the window is equivalent to doing a Reduce to Fit.

## Opening Circuit Page Windows

In DesignWorks, each circuit page is displayed in a separate window that can be opened, closed, scrolled and zoomed independently. The Pages command can be used to display any existing page, as well as adding new pages and setting page order and title. See more information on the Pages command in “Creating a New Page” on page 79.

## Locating Circuit Objects with the Find Tool

The find tool allows you to locate any device, signal or pin by name or other parameter. It will automatically open closed pages and adjust zoom and scrolling to highlight the located objects. To use it:

Select Find in the Tools menu.

Select the type of object you wish to search for, device, signal or pin.

Select the scope of the search, i.e. current page, current circuit level, entire design.

Enter the name of the object to search for, or other search criteria.

Click the Find button.

The Find tool automatically displays the first item found. The Next button can be used to cycle through all objects matching the search criteria.

***More detailed information on the Find tool is provided in "Using the Find Tool" on page 156.***

### Locating Objects Using the Browser Tool

The Browser tool is a convenient way of viewing all objects in the design in tabular format. The Show button on the Browser panel can then be used to display any given object. See "Using the Browser Tool" on page 159 for more information.

## Saving a Design

An open design can be saved to disk using the standard **Save Design** and **Save Design As** commands in the File menu. **Save Design** saves the circuit back into the file that was most-recently opened. It will be disabled if no file has been opened. If you select **Save Design As**, a standard file save box will be displayed requesting the name the new file. The default name will be the current title of the circuit window, i.e. the name of the most recently opened or saved file.

Files can also be saved automatically for backup purposes using the backup commands described in "Backup Procedures" on page 64.

### Reverting to a Saved File

The Revert command in the File menu rereads the current design from the disk file it was last saved to or read from. If any changes have been made since the last Save, you will be prompted to confirm the choice before they are discarded.

## Saving a Circuit Page in PICT Format

DesignWorks can save the circuit diagrams in the Macintosh standard PICT graphics format. This capability allows you to pass graphics to other programs for plotting, enhancement, or incorporation into other documentation.

PICT format is a standard format for representing graphics objects such as lines, circles, text, etc. This format can be read by most Macintosh-based graphics and drafting packages and pen-plotter drivers. PICT file format does not restrict the size of the diagram. Note that DesignWorks cannot read these files, it can only create them.

**Save Page Pict** stores the circuit page in the topmost circuit window to a PICT format file. Note that variations in coordinate systems for different programs may cause the diagram to be shifted off to one side when it is read by another graphics program. This can usually be remedied by selecting everything from within the draw program and centering in the drawing space. The border and background grid will appear or not appear in the file according the “Print Drawing Grid” and “Print Default Border” options in the Custom Sheet Info command.

# Printing

The **Print Design** command allows you to print all or part of your circuit diagram to any printer or other output device that is selectable by the Chooser as a printing device.

## Fitting the Diagram to the Available Paper

DesignWorks has a very flexible ability to set the size of a circuit page, either independently of the current printer, or derived from the printer settings. If, as a result of these settings, the diagram will not fit on a single page, it will be broken into as many parts as are needed, based upon paper size specified in Page Setup. You can preview the page breaks by using the Show Page Breaks option in the Design Preferences command.

This table summarizes the relationship between circuit page size and printed output. More information on these settings is given in “Setting Sheet Sizes and Borders” on page 350. These settings are set using the

Custom Sheet Info command in the Drawing menu.

| Fixed Sheet Size/ Use Page Setup | Auto-Expand | Fit to single sheet | Comments |
|---|---|---|---|
| **Fixed** | On/Off | Off | The circuit page size is fixed and independent of the printer settings. Each page will be broken into as many printed sheets as necessary for output. |
| **Fixed** | On/Off | On | The printer scaling will be adjusted individually for each page so that the page just fits on a single printed sheet. |
| **Use Page Setup** | Off | N. A. | The page border adjusts automatically based on the printer Page Setup settings so a single circuit page always corresponds exactly to a single printed sheet. |
| **Use Page Setup** | On | N. A. | The page border expands in exact multiples of the printed sheet size. |

N. A. = This option not available in this mode.

## Specifying the Page Number Range

For purposes of specifying a range to print, printed sheets are numbered from top to bottom, then left to right, starting on circuit page 1. The page range settings in the standard Print dialog box correspond to printed sheet numbers, not to circuit pages.

In multi-page circuits the print pages are numbered sequentially from the top left corner of circuit page 1 to the bottom right corner of the last circuit page. Page numbers do not appear in the printed output unless they are explicitly placed there using a text variable. See “Using Text Variables” on page 110.

In hierarchical designs, pages are numbered separately for each circuit level. The Print Range command is used to determine printing extent. See more information on printing in hierarchical designs in “Printing Hierarchical Designs” on page 250.

### Setting the Printer Page Setup

The Page Setup command in the File menu allows you to choose the size and orientation of printer paper you wish to use. Once chosen, this information will be stored with your design file and affect the page outlines shown in the command and the Show Printed Page Breaks option in the Design Preferences command.

Depending on the settings in the Custom Sheet Info box, the page setup may affect the displayed page border in your schematic. See “Fitting the Diagram to the Available Paper” on page 62 for more information.

## Backup Procedures

DesignWorks 4.7 has several backup features to allow you to fine-tune your backup strategy to suit your work habits, security requirements and available disk space. These features are controlled by inserting keywords in the setup file, as described in the following sections.

### Enabling Auto-Backup on Save

When auto-backup is enabled, the following sequence of actions is performed every time you select the Save Design command in the File menu:

> The existing design file is renamed to the same name with “.Backup” appended. If the file name is too long to add “.Backup”, characters are removed from the end of the design name until the backup name is short enough. If a file with the “.Backup” extension already exists, it is discarded. See the Checkpoint facility described in “Checkpointing a Design” on page 65 if you want to create a series of backups without discarding older versions.
>
> The design is saved to a new file with the original design name in the same directory.

Note that the “created date” of the new file is set to be the same as the old file so it will retain its historical created date.

The auto-backup feature is enabled by placing the following line in the setup file:

```
AUTOBACKUP;
```

**IMPORTANT: If a disk failure occurs (such as disk full) while the file is being saved, the existing file will already have been renamed to “design.Backup” and will not be automatically switched back to its original name by the program.**

### Enabling Timed Auto-Save

Timed auto-save allows you to request a prompt to save the design every time a specified number of minutes has elapsed. At the appointed time, a box will give you a choice of:

Saving the file. This will also create an auto-backup file, as described above.

Postponing until another interval elapses.

Disabling timed auto-save completely.

Note that if you choose the last option, timed auto-save will be disabled until you quit and restart the program.

Timed auto-save is an extension of the auto-backup feature and is enabled by placing the following line in the setup file:

```
AUTOBACKUP 30;
```

where “30” can be replaced by any desired number of minutes. Note that because timed auto-save is enabled as part of the AUTOBACKUP line in the setup file, there is no way to have timed auto-save without auto-backup on save.

## Checkpointing a Design

The Checkpoint Design command allows you to save known good copies of a design as you work, without having to use a combination of File menu commands and switching out of the program to rename files.

When you select the Checkpoint Design command, the current design will be saved to a file having the same name as the current design with the extension ".Checknnn" added. The "nnn" part of the name will be an integer that will be incremented until an unused name is found. The checkpoint file is saved to the directory containing the current design file.

**NOTE: Checkpoint files are never deleted by DesignWorks. It is the user's responsibility to determine how long to keep these files or whether to move them to a longer-term backup.**

Note that the Checkpoint Design command will be disabled until the design has been stored at least once with a Save As command to establish its directory location.

### Enabling Timed Auto-Checkpoint

Timed auto-checkpoint allows you to request a prompt to checkpoint the design every time a specified number of minutes has elapsed. At the appointed time, a box will give you a choice of:

Checkpointing the file, as described above.

Postponing until another interval elapses.

Disabling timed auto-checkpoint completely.

Note that if you choose the last option, timed auto-checkpoint will be disabled until you quit and restart the program.

Timed auto-checkpoint is enabled by placing the following line in the setup file:

```
AUTOCHECKPOINT 120;
```

where "120" can be replaced by any desired number of minutes.

Note that if a timed Save and a timed Checkpoint come due at the same time, the program will prompt for a Checkpoint, but not a Save. The Save will be postponed until another interval elapses. This is done on the assumption that the Checkpoint interval will be longer and will be a multiple of the Save interval. See more information on this in the next section.

## Backup Strategies

The combination of the timed auto-save and auto-backup features allow you to create a two-tiered backup system. This can help you recover from inadvertent edits, disk failures, program crashes, or revision-control mix-ups with a minimum of lost work. It is best to work out a consistent strategy for backups, including:

The time settings you will use for DesignWorks auto-saves and auto-checkpoints.

How often you will move your design and checkpoint files to backup storage.

How often you will remove the checkpoint files from your main system.

The Save and Checkpoint features are designed with the assumption that auto-checkpoint will be done less often than auto-save. If both operations come due at the same time, the checkpoint takes priority and the save is postponed until its next interval. For this reason, it is most convenient make the checkpoint period an exact multiple of the save period. For example, you may wish to save every 30 minutes but only checkpoint every 4 hours (240 minutes), to avoid creating a large number of checkpoint files.

## Closing a Design

The **Close Design** command in the File menu closes all the circuit windows associated with the current design and removes all data from memory. If the design is hierarchical, all hierarchy levels will be closed. If any changes have been made to your design since the last Open or Save, then you will be asked if you wish to save those changes. The same effect is achieved by clicking the “Go Away” box in the upper left corner of the last circuit window.

### Disposing of a Design

DesignWorks has no built-in command to dispose of a design file. All information about a design is stored in a single file. Dragging the file to the Trash using the Finder has the effect of erasing all data concerning that design.

## Exiting DesignWorks

The Quit command in the file menu closes the DesignWorks application. If any designs are open and have been modified, DesignWorks will ask if you wish to save them.

DesignWorks is also capable of receiving Quit messages from the system, so it will quit in response to selecting a Shutdown command from the Finder and other similar situations.

# Basic Schematic Editing

# 5

The purpose of this chapter is to give you enough basic schematic editing procedures to allow you to create and edit a complete schematic diagram in DesignWorks. Before you start working on a major design, though, we do suggest that you review Chapter 6—Before Starting a Major Design on page 117. That chapter will look at interfacing with PCB and simulation systems, generating hardcopy output and other issues. Giving those points a bit of thought before you start can save you a lot of headaches later. This chapter will also point you to later chapters for more in-depth information on specific topics.

## General Editing Concepts

### Undo and Redo

The Undo command undoes the last editing operation that was performed. The text of this menu item will change based on the type of operation. Generally, only schematic editing operations can be undone. Major structural changes, such as Define Attribute Fields, adding pages, or any menu commands involving a dialog box are usually not undoable. DesignWorks stores up to 10 levels of Undo information.

Undo never changes the contents of the Clipboard. E.g. after a Cut operation, Undo will restore the schematic, but leave the Cut objects on the Clipboard.

The Redo command redoes the last Undo command. It will only be enabled immediately after an Undo operation. Any other editing operation will disable this item.

### The Clipboard

The standard Macintosh clipboard commands Cut, Copy and Paste can be used to move or copy circuit fragments, graphical and text information within a single circuit window, between multiple windows, and between different programs (e.g. word processing or drafting).

### Using Clipboard Data From Other Programs

When you enter DesignWorks, the clipboard may contain graphical or text information Cut or Copied from a document in another program. DesignWorks allows you to make use of this information in two ways:

> A Macintosh picture can be Pasted directly onto your diagram in any desired position. Once placed, it can be dragged, Zapped, Cut, Copied, Duplicated, etc. using the various editing commands available for other circuit elements. When a picture item is selected, the Get Info command can be used to select an optional border. A graphics object is considered to be a monolithic item on a schematic and can only be moved or deleted in its entirety. It can, however, be copied and pasted to the DevEditor tool to take advantage of the standard drawing tools available there.
>
> Text information from a word processor or text editor can be pasted into a text block. See more information under the Text command below.

### Using Clipboard Data From DesignWorks

When a Cut or Copy is done, two types of data are placed on the clipboard:

> A Macintosh picture (“PICT 2” format) of the selected items, which can be pasted into a graphics document using most drawing programs.
>
> The DesignWorks circuit info for the selected items. This data is in a format that only DesignWorks can understand and is discarded when you Quit.

**WARNING: Circuit structural information on the clipboard is discarded when you quit the program. Only picture and text data is retained. You cannot Copy and Paste circuit data between DesignWorks sessions.**

Cut and Copy work on the currently selected group of circuit objects and will be disabled if no objects are selected. When items are copied onto the clipboard, their names are copied with them, which may result in duplicate names. If duplicate signal names are pasted back into the circuit page they were copied from, then logical connections will be made between the like-named segments.

### Using the Clipboard Commands

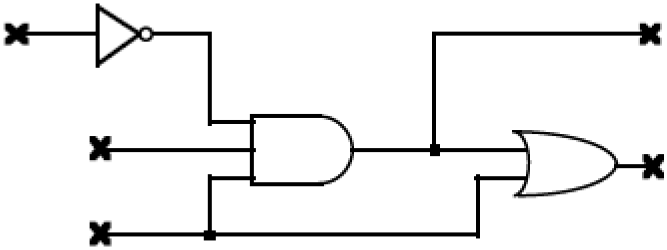
The standard clipboard commands Cut, Copy and Paste commands are available for editing, with these characteristics:

| | |
|---|---|
| **Cut** | Cut removes the currently selected objects from the circuit and transfers them to the clipboard. It is equivalent to doing a Copy and then a Clear. Cut will be disabled if no objects are selected. |
| **Copy** | Copies the currently selected objects onto the clipboard without removing them. This can be used to duplicate a circuit group, copy it from one file to another, or to copy a picture of the circuit group to a drawing program. See the notes on clipboard data above. Copy will be disabled if no objects are currently selected. |
| **Paste** | Paste replaces the cursor with a flickering image of the contents of the Clipboard in the circuit window. The data being pasted may be a circuit group copied from within DesignWorks, or it may be text or picture information created by another program or module. The image of the clipboard data can be dragged around and positioned as desired. The item will be made a permanent part of your diagram when the mouse button is pressed. Paste will be disabled if there is no information on the clipboard of a recognized type. |

**TIP: As well as the standard location in the Edit menu, there are also Cut, Copy, Paste, Clear and Duplicate commands in the Device and Signal pop-up menus. These operate on the single object that was clicked on to display the menu.**

### Signal Connection Checking on Paste

DesignWorks checks for signal connections only at “loose ends” in the signal lines being pasted, i.e. ends of line segments which do not touch devices or other line segments. E.g. if the following circuit scrap was pasted, the points marked X would be checked for connection to the existing circuit.



**NOTE: Connection "hit testing" can be disabled by holding the Option key on the keyboard depressed while clicking the mouse button (this also applies to single device placing). In this case, the circuit scrap is placed, but no connections will be made to adjacent items. This allows the group to be selected again (by Option-double-clicking on any device in the group) and moved without interactions with other objects in the circuit.**

### Duplicating One or More Objects

The Duplicate command makes a copy of the selected circuit group which can be dragged and positioned as desired. This is equivalent to selecting Copy and then Paste, except that the selected circuit scrap is not placed on the clipboard for future use. See the notes under Paste, above, on how connections are made when a group is placed in the circuit. Note that the duplicated objects can be rotated using the orientation pop-up menu in the tool palette or the arrow keys on the keyboard.

### Rotation on Paste and Duplicate

Any group of objects being Pasted or Duplicated can be rotated using the same controls as when placing a device:

The Orientation pop-up menu in the tool palette.

The arrow keys on the keyboard.

Note that these controls are only effective while actually moving the flickering image of the object being pasted. Each Paste or Duplicate always starts in the same orientation as the source.

**NOTE: The Orientation command cannot be used for Paste or**

**Duplicate since selecting this menu command will abort the paste operation.**

## Selection

Many DesignWorks commands, such as Get Info, Cut, Copy, etc., operate on the currently selected objects. In general, any object can be selected by clicking on it, but a few exceptions exist and numerous other convenience features are available. For a detailed list of the techniques available for selecting objects, see the following procedure sections in this chapter.

### Selecting Groups of Objects

Several methods are available for selecting multiple objects:

1) Any group of adjacent items can be selected by clicking and dragging across the group. A flickering rectangle will follow the mouse movement. Any object that intersects this rectangle when the button is released will be selected (except background objects).

2) A group of interconnected devices and signals is selected by double-clicking on any device in the group while holding the Option key pressed. If a circuit is completely interconnected, this will select the entire circuit.

3) The Select All command in the Edit menu selects all items on the current page except background items.

***See "Selecting a Background Object" on page 114 for more information on background objects and how to select them.***

4) The SHIFT key can be used in combination with any of the above methods to select multiple items. When the SHIFT key is held, the previously selected items are not deselected when a new item is clicked on. Thus you can add to the selected group until the desired collection of items is selected.

### Restricting Object Types while Selecting

While performing any click or drag-selection operation, you can restrict which types of objects get selected by holding down one or more keys on the keyboard corresponding to the object types desired. This can be used to select an object that is completely covered by another, e.g. a pin that is buried in a device symbol. Here are the object type selections available:

| **Keyboard Key** | **Object Type Selected** |
|---|---|
| D | Device |
| P | Pin |
| S | Signal |
| T | Text |
| G | Picture (graphics) |

For example, if you wish to select all the pins on a device, but not the device itself, you can hold the P key on the keyboard depressed while drag-selecting across the device. You can also hold multiple keys simultaneously to select multiple types of objects. For example, to select all text and picture objects on a page, you could hold the T and G keys depressed and select the Select All menu item. Note that you won't be able to use Command key equivalents in conjunction with this mechanism.

### Deselecting a Selected Object

All currently selected objects are deselected by clicking in an empty area of the schematic window. A single item can be deselected by holding the SHIFT key while clicking on it.

### Multi-page Selection

The DesignWorks commands that operate on selected objects normally only affect objects on the current page. However, some external tools, such as the Browser and Find, operate across pages and can select objects on non-current pages. This is done to allow global operations to be done without having to go to each page and perform separate editing operations. For example, Find can be used to locate all devices of a particular type and select them, then Browser can be used to display and edit their attributes.

Note the following rules for multi-page selection:

Clicking in an empty part of a schematic window deselects all objects everywhere in the design.

Switching from one schematic window to another does not change the status of any objects that were selected on either page.

You can select objects on one page, switch to another page, and (using the normal SHIFT key technique) add items on the new page to the selected group.

Editing commands like Cut, Copy and Clear will only affect selected items on the current page, even if objects are selected on other pages.

## Zooming in on Selected Objects

The Go To Selection command causes the circuit position and scaling to be adjusted so that the currently selected items are centered and just fit in the circuit window. The scaling will be set to 100% maximum. This command works even if the selected objects are on a page other than the current one.

## Adjusting the Position of All Objects on a Page

If you wish to adjust all the objects on a page to make room for editing or to re-center an edited drawing, you can:

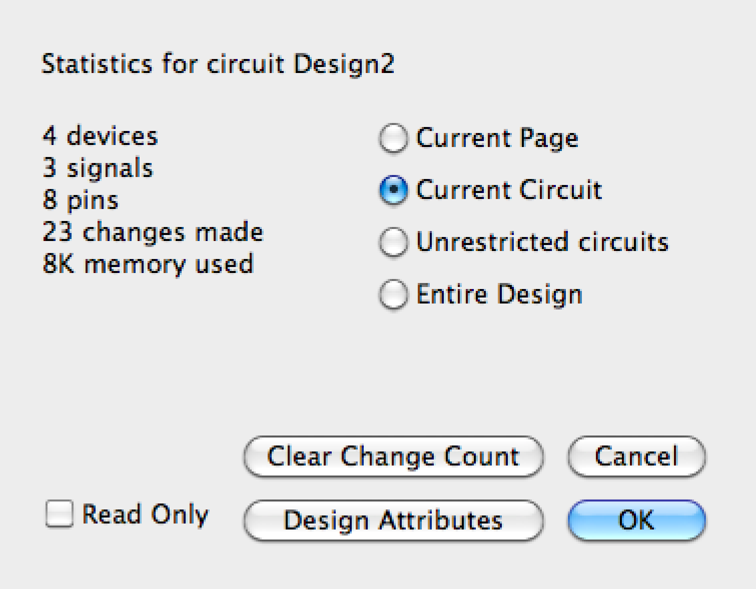
Use the Select All command in the Edit menu to select all the objects on the page, then click and drag inside any device symbol to move the entire group.

Use the Center in Page command in the Drawing menu. This command moves all items on the current page (except text and picture objects marked as "border" items) so that the circuit objects (taken as a group) are centered in the page. If the "Auto-Expand" option is turned on in the "Custom Sheet Info" box, then this command may result in the number of pages required being reduced.

***More information on the Custom Sheet Info command can be found in "Setting Sheet Sizes and Borders" on page 350.***

## Showing Overall Circuit and Design Statistics

If no objects are selected in the circuit (i.e. if you have clicked in an empty portion of the diagram) then Get Info will display this general design information box:



Statistics for circuit Design2

4 devices
3 signals
8 pins
23 changes made
8K memory used

○ Current Page
● Current Circuit
○ Unrestricted circuits
○ Entire Design

☐ Read Only

The buttons on the right hand side of this box allow you to chose the scope of the displayed counts:

| | |
|---|---|
| **Current Page** | This selection shows information only on objects displayed in the circuit page displayed in the frontmost window. |
| **Current Circuit** | This selection shows information on objects on all pages of the current circuit level. |
| **Unlocked Circuits** | This selection shows information on objects in the master circuit of the design and all sub-circuits not marked as “Locked” in their respective Device Info boxes. |
| **Entire Design** | This selection shows information on all circuits in the design. |

The following items of information are shown for the selected scope:

| | |
|---|---|
| **nnn devices** | This is a count of devices in the selected scope. Pseudo-devices, such as page connectors and breakouts are not included. In "Current Page" or "Current Circuit" mode, this count includes devices that have sub-circuits. In "Unlocked Circuits" and "Entire Design" mode, this count includes only "bottom-level" devices, i.e. those without a sub-circuit or whose sub-circuit is not being listed. |
| **nnn signals** | This is a count of signal nets in the design, including unconnected pins. Signals that pass through more than one hierarchy level are included once for each level that they exist in. |
| **nnn pins** | This is a count of device pins, not including pseudo-devices. |
| **nnn changes made** | This is a count of editing changes made since the design was created. This is intended to allow comparisons of different versions of the same file. This value can be cleared using the Clear Change Count button and can be extracted in reports using the $CHANGECOUNT script keyword. |
| **nnnK memory used** | This is a count of the amount of main memory occupied by the selected part of the design, in Kbytes. |

## Setting Design Attributes

To edit text attributes associated with the design, you need to display the general design attribute editing box. This can be done in either of these ways:

Click in an empty area of the schematic to ensure that no objects are selected, then choose the Get Info command in the Options menu. This will display the circuit info box already described. In this box, click on the Design Attributes button.

Select the Set Design Attributes command in the Options menu.

***For more information on the usage of attributes, see "Entering and Editing Attribute Data - Basic Procedure" on page 175.***

## Making a Circuit Read Only

You can make a circuit read-only to prevent accidental modification to it. This is done by following these steps:

Click in an empty area of the schematic to ensure that no objects are selected.

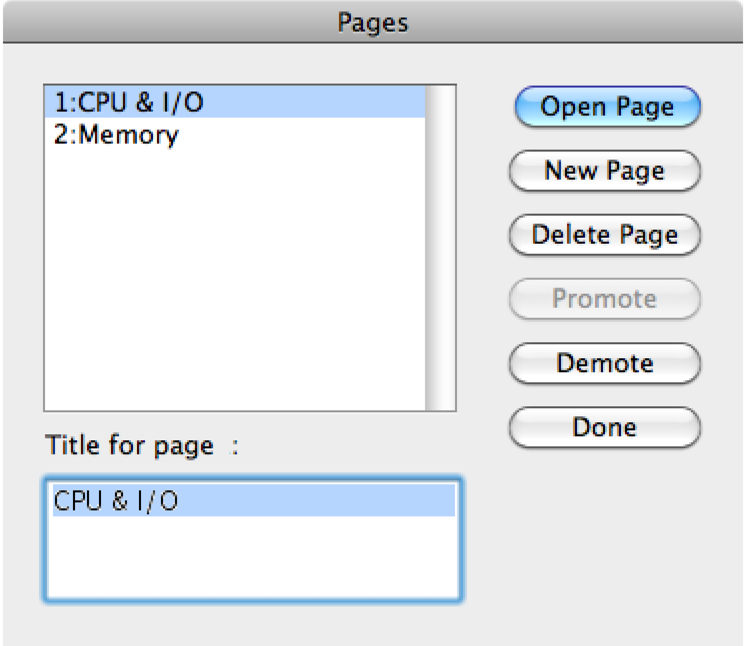
Choose the Get Info command in the Options menu. This will display the circuit info box already described.

In this box, turn on the Read Only switch. Note that this switch is enabled only when the “Current Circuit” button is selected since it only applies to one circuit level.

In a hierarchical design, the “Read Only” status applies only to the current circuit level in a hierarchical design and to all levels below it. Note that “Read Only” is not the same as “Locked” in that it allows you to view the circuit but not change it.

## Adding, Deleting and Titling Circuit Pages

The Pages command is used to create a new page, display an existing page, change page order, or set a page title. Executing this command will produce the following box:



1:CPU & I/O
2:Memory

Title for page :

CPU & I/O

Operations on pages are performed by selecting the pages in the scrolling list and clicking on one of the operation buttons. More than one page can be selected at a time (for all operations except setting the page title) by holding the command or shift keys pressed while clicking in the list.

**WARNING: Operations in this box cannot be undone!!!**

### Opening a Page

The Open Page button causes the pages selected in the list to be displayed in a window. If a window is already open with the selected page, then it is brought to the front.

### Creating a New Page

The New Page button causes a new, empty page to be added to the circuit and opens it in a new window.

### Deleting a Page

The Delete Page button causes the page selected in the list to be deleted from the circuit. Note that the page must already be empty, i.e. all objects must have been removed from the page (for example using the Select All and Clear) commands before this will be allowed. Higher numbered pages will be renumbered when this page is removed.

### Changing Page Order

The Promote and Demote buttons allow you to change the number order of the pages. Promote causes all pages selected in the page list to be moved one step toward the front of the list, i.e. their page number will be reduced by one. All non-selected pages will be renumbered as needed. Demote similarly moves all selected pages toward the end of the list. Promote will be disabled if the first page is selected and Demote will be disabled if the last page is selected.

### Setting a Page Title

The page title text box allows a title of up to 63 characters to be entered for the selected page. This title can be accessed via the $PAGETITLE variable in text boxes. See “Using Text Variables” on page 110.

# Working with Device Symbols

## Placing a Device From a Library

To select a device from a library for placement in the schematic:

Select the desired library using the pop-up menu at the top of the Parts palette. The palette displays only the contents of only one library at a time. If the library file you need is not open, you can open it using the Open Lib command in either the Libraries sub-menu of the File menu, or in the Parts palette pop-up menu.

If necessary, use the scroll bar to scroll the list until the desired part name is in view.

Double-click on the part name in the list.

Move the cursor to the current schematic window.

### About Device Libraries

The symbols and related parameters for DesignWorks devices are stored in data files called symbol libraries. Libraries can be opened and closed using the Open Lib and Close Lib commands, or using entries in the setup file.

For each device type in a library the following data is stored:

general information on the type, such as number of pins, number inputs, number of outputs, type name, default delay, default attributes, position, orientation and type of each pin, etc.

a picture representing the symbol for this type.

a polygon outlining the symbol, used for highlighting and erasing the symbol.

an optional internal circuit definition.

### How Device Symbols are Created and Stored

Libraries are created and modified using the DevEditor tool, which is described elsewhere in this manual.

***See Chapter 12—Device Symbols on page 271 for more information.***

The cursor will be replaced by an image of the selected device. While moving this image around, you can use the arrow keys on the keyboard or the orientation tool on the tool palette to rotate the symbol.

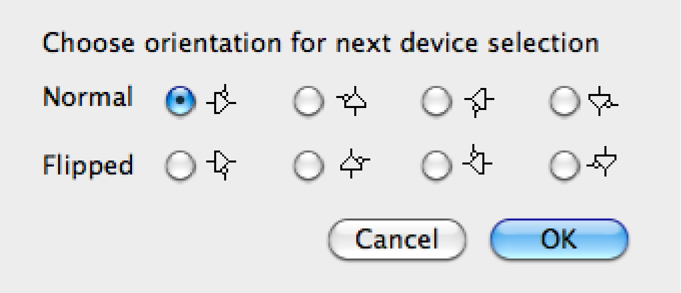
Clicking anywhere in the circuit diagram will make a permanent copy of the device at that point.

**TIP: Holding the Option key while clicking will inhibit checking for pin connections. This allows you to select the device again and drag it to a new position without affecting any existing connections.**

### Setting Device Orientation

Device symbols can be placed in any of 8 different orientations, that is, the 4 major compass points, plus a 180 degree flip around each of those axes. The orientation that a symbol will be placed in is determined by the current orientation setting. This can be controlled in any of the following ways:

The Orientation command sets the orientation (up, down, left, right, mirrored) that will be used next time a device is created. When this command is selected, the following box is displayed:



Choose orientation for next device selection

Normal ● ○ ○ ○

Flipped ○ ○ ○ ○

The orientation can be set by clicking directly on the orientation icon in the tool palette. This displays a pop-up list of orientation icons that you can choose from.

The orientation ca be set using the arrow keys on the keyboard. Press any arrow key once to select that direction. Press the same key again to flip the symbol around that axis.

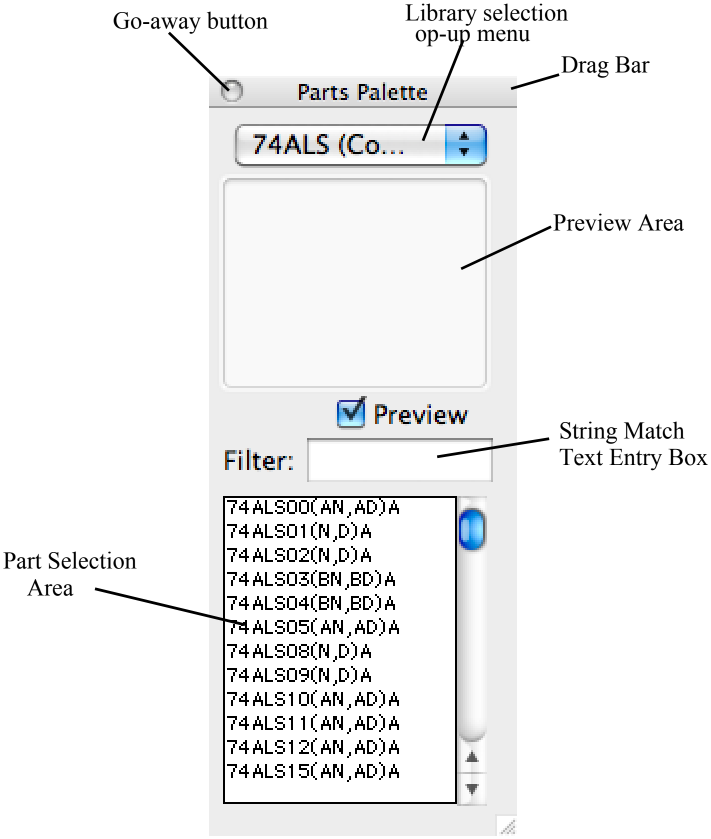
**TIP: Selecting any menu item cancels the device placement mode, so either the tool palette or the arrow key method is**

**preferred since it allows you to change orientation while placing a device or circuit scrap.**

***You can also flip and rotate a device symbol after it has been placed in the schematic. See "Flipping and Rotating a Device" on page 85.***

## Using the Parts Palette

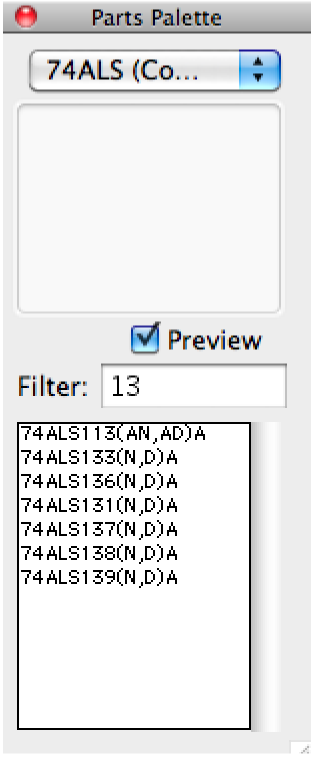
Parts library contents are displayed in a floating palette window which looks like this:



The Parts palette displays the contents of the selected library file and allows any item to be selected for placement in the schematic.

## Using the String-Matching Box

The string match text box allows you to type characters which will reduce the size of the list and make it easier to locate the desired part. Simply click in the text box and type the desired characters. After a brief pause, the displayed part list will be reduced to only those parts that contain the typed string of characters. For example:



Filter: 13

74ALS113(AN,AD)A
74ALS133(N,D)A
74ALS136(N,D)A
74ALS131(N,D)A
74ALS137(N,D)A
74ALS138(N,D)A
74ALS139(N,D)A

To return to the full selection, click at the end of the characters in the box and backspace over them until the box is cleared.

## Parts Palette Pop-up Menu

As a shortcut, the Libraries sub-menu in the File menu can be popped up right in the Parts palette. Hold the Control key on the keyboard while clicking and holding anywhere in the Parts palette window. Menu selections made here are equivalent to selecting the same item in the Libraries sub-menu.

***See "Working With Symbol Libraries" on page 272.***

### Moving the Parts Palette

The Parts palette can be moved to any desired location on the screen by clicking and dragging in its drag bar.

### Hiding the Parts Palette

The Parts palette can be removed from the screen by clicking in its "go-away" box. To re-display the palette, select the Show Parts Palette command in the Libraries sub-menu of the File menu.

## Duplicating an Existing Device

To duplicate and existing device, either:

> Select a similar device anywhere on the current circuit page and use the Duplicate command (either in the Edit menu or in the device pop-up menu).
>
> Select a similar device in any other page of any open design and use the Copy command. Return to the destination circuit and select the Paste command.

After either of these operations, the cursor will be replaced by an image of the selected device which can be placed by clicking in the schematic, as discussed above.

## Deleting a Device

Devices can be removed by either of these two methods:

> Select the device by clicking on it (holding the SHIFT key if it is a switch or other input device) and then hit the "delete" or "delete" key on the keyboard, or select the Clear command from the Edit menu.
>
> Enter Zap mode by selecting the Zap command or clicking on the Zap icon in the tool palette, then click on the device in question.

## Moving a Device

Devices can be moved by clicking and dragging to the desired new position. If more than one device is selected, all the devices, and all signals connecting between them (whether or not selected) will be moved. Signal lines will be adjusted to maintain right angles at points where moving signal lines intersect with non-moving ones.

## Flipping and Rotating a Device

A set of four commands for flipping and rotating an existing device symbol can be found in the device pop-up menu, i.e. by holding the Control key while clicking on the device. The Rotate Left/Right and Flip Vertical/ Horizontal commands are equivalent to deleting the selected device and replacing it in the selected new orientation. If the new rotation causes any device pins to touch adjacent signal lines, connections will be made unless the Option key is held.

***When placing a device symbol, you can determine its orientation in advance using the tool palette or the arrow keys on the keyboard. See "Setting Device Orientation" on page 81.***

## Displaying and Setting Device Information

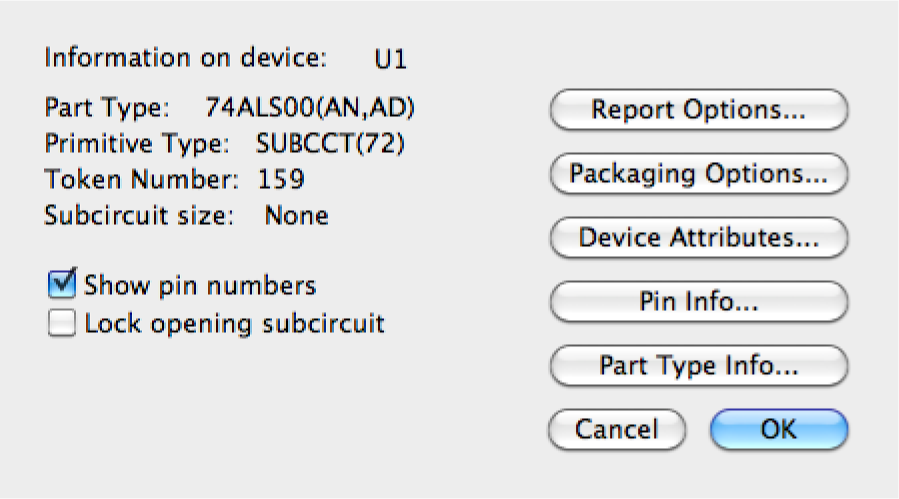
The device information box allows you to view and set a lot of vital information about the selected device. To display the device information box, either:

Hold the Control key while clicking on the device, then select the Device Info command in the pop-up menu, OR,

Select the device by clicking on it, then choose the Get Info command in the Options menu.

Quite a few options are available from this box, as summarized in this table:

When a normal device symbol is selected on the schematic (i.e. not a pseudo-device), then the following information box is displayed:



Information on device: U1

Part Type: 74ALS00(AN,AD)

Primitive Type: SUBCCT(72)

Token Number: 159

Subcircuit size: None

☑ Show pin numbers

☐ Lock opening subcircuit

The following table lists the information and options available in this box.

| | |
|---|---|
| **Part Type** | This is the library type name of the device symbol, i.e. the name as it appears in the Parts palette. This is not the same as the Part attribute field, which is normally used as the part name in netlists. |
| **Primitive Type** | This is the primitive type of the symbol. For standard types, the name is shown, otherwise the name “Reserved” is shown. The ordinal number of the primitive type value is shown in parentheses. This is normally only of interest in specialized applications. |
| **Token Number** | This is a permanent number that is assigned to this device instance for use in internal DesignWorks operations and some back annotation and other interface operations. More information on tokens can be found in “Device Token Values” on page 151. |
| **Subcircuit Size** | If the selected device has a sub-circuit, its memory size is shown in Kilobytes. |
| **Show Pin Numbers** | This switch allows you to disable the display of pin numbers for the entire device. This is intended for discrete components or others where pin numbers are not normally shown on the diagram. |
| **Lock Opening Subcircuit** | This switch allows you to prevent the sub-circuit (if any) of this device block from being opened for editing. See “Locking and Unlocking Subcircuits” on page 242 for more information. |

| | |
|---|---|
| **Report Options** | This button allows you to select whether this device or its internal circuit will be listed in a netlist. See “Device Reporting Options” on page 371 for more information. |
| **Pin Info** | This button displays the Pin Info box for the first pin on the device. Buttons on the Pin Info box allow you to sequence through the other pins on the same device. See “Getting and Setting Pin Information” on page 106 for more on the Pin Info box. |
| **Part Type Info** | This displays the part type info box, which provides more information on the type definition used for this device and allows you to locate other devices of the same type. More information on this option can be found in “Displaying Part Type Information” on page 87. |
| **Packaging Options** | This button displays the current packaging level for this device. The meanings of these settings are described in “Setting Device Packaging Options” on page 139. |
| **Device Attributes** | This button displays the standard attribute edit box for the device. See “Entering and Editing Attribute Data - Basic Procedure” on page 175 for more information. |

**NOTE: Clicking Cancel on the Device Info box <u>does not</u> cancel changes that were made in other boxes displayed using Device Info option buttons.**

## Entering Device Attributes

To enter device attribute, either:

Hold the Control key while clicking on the device, then select the Attributes command in the pop-up menu, OR,

Select the device by clicking on it, then choose the Get Info command in the Options menu, then click the Attributes button.

***See more information on attributes in Chapter 9—Attributes on page 173.***

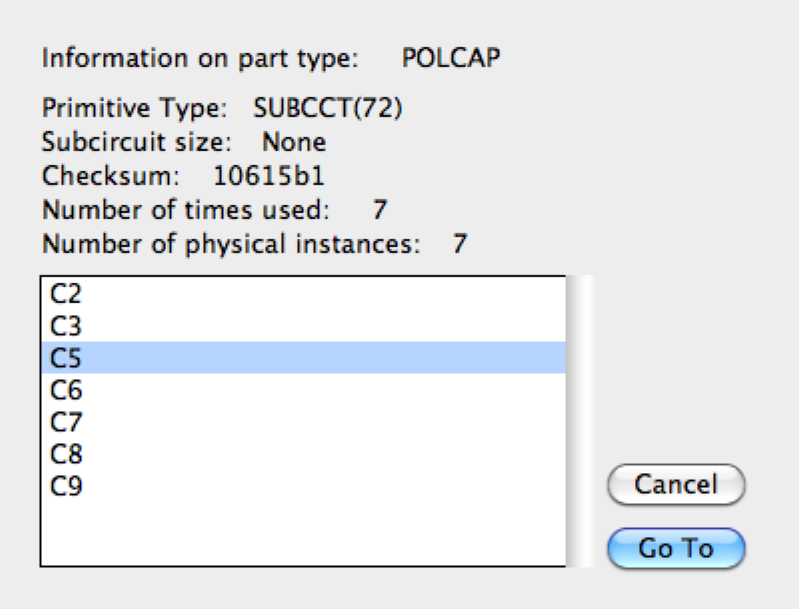
## Displaying Part Type Information

To display more information on the part type (i.e. symbol definition) associated with a device:

Display the device info box using the methods described in “Displaying and Setting Device Information” on page 85.

Click on the Part Type Info button.

This will display an information box like the one shown here:



Information on part type: POLCAP

Primitive Type: SUBCCT(72)
Subcircuit size: None
Checksum: 10615b1
Number of times used: 7
Number of physical instances: 7

C2
C3
C5
C6
C7
C8
C9

Here is a description of the options and information in this box:

| | |
|---|---|
| **Primitive Type** | This is the primitive type of the symbol. For standard types, the name is shown, otherwise the name "Reserved" is shown. The ordinal number of the primitive type value is shown in parentheses. This is normally only of interest in specialized applications. |
| **Subcircuit Size** | If the selected device has a sub-circuit, its memory size is shown in Kilobytes. |
| **Checksum** | This value is used internally to distinguish when two like-named symbol definitions are identical. Any symbol editing operation updates this value. This value can be extracted in reports using the $CHECKSUM script keyword. |
| **Number of times used** | This is the number of times the symbol appears in circuits, counting a given subcircuit block only once. |
| **Number of physical instances** | This is a count of the number of physical incarnations of this device exist in the design, counting all instances of parent hierarchy blocks that may contain this symbol. In Pure or Flat hierarchy mode designs, this will be the same as "number of times used". |
| **Instance List** | The list shows all usages of this part type in the design. Double-clicking on an item will close this box and display the selected item, if possible. |
| **Go To** | This displays the device instance selected in the instance list. |

## Device Names

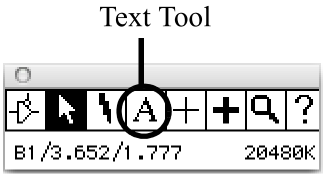
**NOTE: In this manual, we use the term "device name" to refer to the character string that identifies a unique device in the circuit. Typical device names might be "U23", "C4", "IC12A", etc. This is distinct from the "type name" or "part name" that is used to distinguish the type definition that is read from a device library. Typical part names are "74ALS138", "MC9S12C128CFUE", etc.**

Device names may contain any letters, numbers or special characters that you can type on the keyboard, but are restricted in length to at most 15 characters. The name associated with an object can be placed anywhere on the diagram and will be removed if the object is removed.

**NOTE: Although DesignWorks does not enforce restrictions, we recommend not using blanks or special characters in device or signal names. This can result in problems exporting data to external systems such as PCB or simulation tools. See Chapter 6—Before Starting a Major Design on page 117 for more information on naming conventions.**

### Adding a Device Name by Typing on the Schematic

Enter "Text" mode either by selecting the Text command in the Edit menu, or by clicking on the text icon in the tool palette:



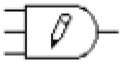
Note that once "Text" is selected, the cursor changes to a pencil icon. Two techniques are available to determine the position of the name:

If you click the pencil cursor on a device and release it immediately, the flashing text cursor will jump to the default name location for that symbol. You can then type the name as desired on the keyboard, ending with the Enter key.

**NOTE: A default position can be specified for any attribute use the methods described in “Using Default Position Fields” on page 190. If no position is specified, the program will calculate one.**

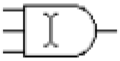
If you wish to determine the starting position of the text before you type it, press and <u>hold</u> the mouse button with the tip of the pencil positioned inside a device symbol. As long as you hold the mouse button down an I-beam cursor will track the mouse movements. The device name text will start at the position where you release the button. Type the desired name and press Enter, or click the mouse button anywhere. This sequence of steps is illustrated below.

1



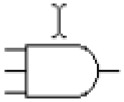
Position the pencil cursor anywhere inside the device symbol.

2



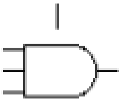
Click and hold the mouse button. The cursor changes to an "I-beam".

3



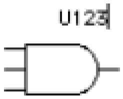
Still holding the mouse button pressed, position the cursor where you want the name to start.

4



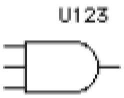
Release the mouse button. A blinking insertion marker will appear.

5



Type the desired name, up to 15 characters.

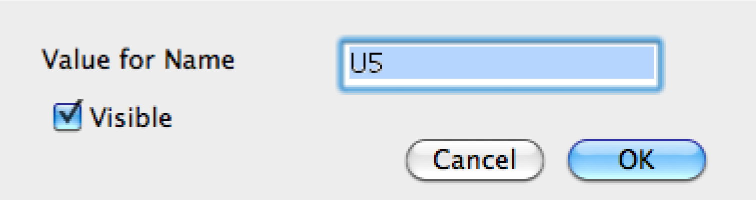
6



Press the Enter key, or click the mouse button once to make the name permanent.

## Adding a Device Name Using a Menu Command

The name can be set by holding the Control key while clicking on the device, and selecting the Name command out of the device pop-up menu. This command displays a simple edit box allowing you to enter or edit the device name. The “Visible” option in this box allows you to select whether the name is made visible on the schematic or simply kept as an invisible text attribute of the device.



Value for Name U5

☑ Visible

### Repositioning or Removing a Name

Once a name is placed, it can be repositioned by dragging it using the Point cursor or removed using the Zap cursor. The device name will be removed automatically if the device is removed. Holding the Option key depressed while moving a name (or any attribute value) disables the grid snap, allowing you to micro-position the name for alignment with other graphic items on the schematic.

**NOTE: The "Name" is actually an attribute field, so all attribute editing techniques can be used on it. In particular, you can hold the Control key while clicking on the name and the attribute pop-up menu will appear, giving you a variety of editing commands, described "Other Ways of Viewing and Editing Attributes" on page 183.**

### Selecting a Device

To select a device, the cursor must be in the normal pointer ( ↖ ) mode. A single device is selected by clicking the mouse button with the pointer positioned anywhere inside the device symbol or in any displayed attribute value associated with the symbol.

If the DesignWorks Simulator option is installed, simulated input devices such as switches and keyboards can only be selected by holding the SHIFT key while clicking. This is necessary because a normal click is used to change the state of these devices when a simulator tool is active.

## Selecting the Part and Package Type

If you place a symbol that has the standard Part and Package attribute fields predefined in it, these text annotations will appear next to the symbol. Many of the standard symbols provided with DesignWorks have several possible Part names for one symbol, corresponding to various package types. To select one of the values for Part, you can either:

Use the device Pop-Up menu. This is done by holding the Control key down and clicking and holding the mouse button with the cursor over the device. Move to the bottom of this menu where a sub-menu with t allowable values for the Part attribute are shown. A new value is set by selecting a unit from the Part menu item. This sub-menu will only appear if the symbol has multiple values specified in the Part.List attribute field.

Use the device attributes box to enter a new value for the Part attribute. One way to do this is to select the device by clicking on it, select the Get Info command in the Options menu, then click on the Attributes button. Select Part item in the field list and enter the desired new value.

In the standard DesignWorks libraries, the Package code is automatically linked to the Part type. In other words, when you select one of the alternate Part values from the list, the Package code is automatically updated to match.

***For more information on the Part, Package and other attribute fields, see "Selecting the Part and Package Type" on page 91.***

## Selecting the Gate Unit

If you place a gate-type symbol (i.e. one that has multiple logical units per package) from the standard DesignWorks PCB libraries, a gate unit will be assigned automatically and the unit value will be displayed after the name, e.g. U23b. When you see such a value, there are actually two attribute fields displayed side by side. In this case, the "U23" portion is the Name field and the "a" portion is the Unit. The Unit will only be shown for symbols that have gate packaging information in them.

**NOTE: If Auto Packaging is enabled (it normally is for PCB designs) then you should not manually change Unit settings. See "Using Device Packaging" on page 131 for more information.**

To change the values for the Unit you can either:

Use the device Pop-Up menu. This is done by holding the key down and clicking and holding the mouse button with the cursor over the device. Move to the bottom of this menu where a sub-menu with t allowable values for the Unit attribute are shown. A new value is set by selecting a unit from the Unit menu item.

Use the device attributes box to enter a new value for the Unit attribute. One way to do this is to select the device by clicking on it, select the Get Info command in the Options menu, then click on the Attributes button. Select the Unit item in the field list and enter the desired new value.

Once you select a new Unit, the pin numbers on the device will be updated to match. If you select a Unit value that is already in use, you will be given a choice of swapping with the other unit, forcing the new value, thus creating an error, or canceling the operation.

**NOTE: This procedure works only for symbols that have the correct packaging attributes set up in advance. For more information on these settings, see "Creating a Symbol with Multi-gate Packaging" on page 142.**

## Creating and Editing Signals

In DesignWorks, a signal represents the electrical connection between any number of device pins. A signal can simply be represented on a schematic by a single line or a number of connected line segments, or more complex structures such as connection by name, busses, cross-page connectors and hierarchy blocks can be used to simply the representation of large designs. In this chapter, we will look at simple signals and connection by name within a single circuit page. Busses and multipage signal interconnection schemes are covered in Chapter 10—Making Signal Connections on page 207.

### Interconnecting Signals

If you draw a signal line such that the end of the line contacts a second signal line, then those two signals will be interconnected. Also, if you place a new device such that one of its pins contacts an existing signal line, that pin will be connected to the signal. If both of the two signals being connected were named, then you will be prompted to choose the name of the resulting signal. Whenever three or more line segments belonging to the same signal meet at a given point, an intersection dot will be placed at that point automatically.

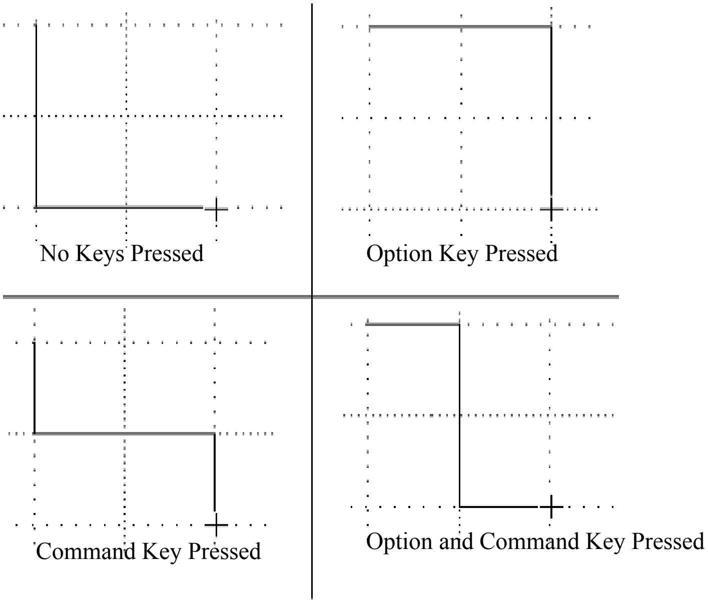
**NOTE: For efficiency, signals are only checked for connections at their endpoints and only signals actively being edited are checked. It is possible to create overlapping lines that do not connect by unusual combinations of editing operations. This situation is usually visually apparent at the time the editing is done since the intersection dot will be missing and the entire signal will not highlight when clicked on. You may wish to also use the ErrorScript tool to locate these situations.**

***See more information on connection checking in "Signal Connection Checking on Paste" on page 71.***

### Drawing From an Existing Line or a Device Pin

A line can be extended from the end of an existing line or device pin using the normal ( ↖ ) cursor. Click and hold on the end of the pin and drag away from the pin. A pair of right-angle lines will follow the cursor away from the pin and long as the mouse button is pressed. Releasing the mouse button makes these lines permanent. If the end of the line (i.e. the point where the mouse button was released) touches another signal line, a connection will be made at that point.

Alternate line routing methods can be activated by pressing the Option and Command keys, as follows:



No Keys Pressed

Option Key Pressed

Command Key Pressed

Option and Command Key Pressed

The Shift key constrains the movement to a single vertical or horizontal line. The Option key inverts the order of line drawing, and the Command key switches to three line segments with a center break.

**NOTE: Holding the Option key while clicking will inhibit checking for pin connections. This allows you to select the signal again and drag it to a new position without affecting any existing connections.**

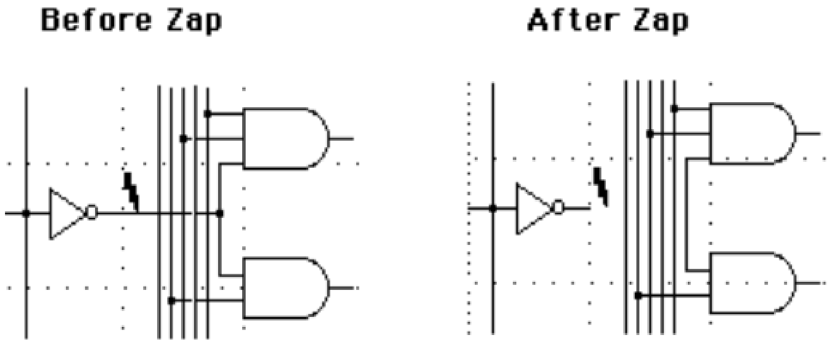
### Creating an Unconnected Signal Line

The Draw Sig (+) cursor can be used to create an unattached signal line, or can be used to extend an existing signal. Simply click anywhere in the schematic and drag away in the desired direction. Unlike the Point mode drawing method, above, the mouse button does not have to be held while creating signals in this mode. Double-clicking terminates the signal line.

### Editing a Signal Line

The following features are available to edit signal lines:

Zap mode (entered by selecting the Zap command in the Edit menu or the Zap item in the tool palette) allows you to remove any single line segment from a signal connection. Zapping on a signal line removes only the line segment being pointed at, up to the nearest intersection, device pin or segment join point.



Selecting a signal line (by clicking anywhere along its length) and hitting delete or selecting the Clear command removes an entire signal trace.

Drawing backwards along the length of an existing line causes the line to be shortened to end at the point where you let the button go.

Clicking and dragging the middle of a signal line segment with the pointer cursor allows you to reposition the line. Vertical lines can be moved horizontally and vice versa.

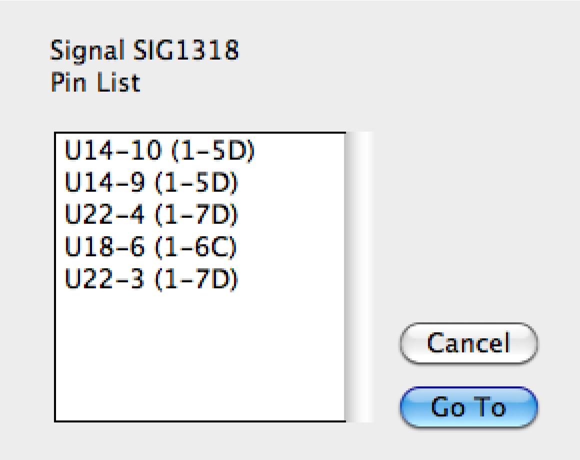
The pointer cursor can be used to start drawing from the ends or corners of an existing signal.

The Draw Sig (┼) cursor can be used to start drawing from anywhere along an existing signal line. Double-clicking terminates drawing.

## Checking Signal Interconnection

Double-clicking anywhere along a signal line will cause that signal segment and all logically connected segments on the current page to be selected.

You can also use the Pin List command in the signal pop-up menu (by Control-clicking on the signal) to view the list of pins attached to the signal and go to any of them. This command displays a list of all device pin connections comprising the selected signal. Only pins in the current circuit are listed; pin connections in other hierarchy levels are not shown.



Signal SIG1318
Pin List

U14-10 (1-5D)
U14-9 (1-5D)
U22-4 (1-7D)
U18-6 (1-6C)
U22-3 (1-7D)

Selecting any item in the pin list and clicking on the Go To button will cause the appropriate page to be opened (if necessary) and that pin to be displayed in the center of the circuit window. If the selected pin is invisible (e.g. a bus internal pin) the window will be centered on the parent bus pin.

For convenience in navigating a schematic, connections to pseudo-devices such as bus breakouts and ground symbols are also shown in this list. In this sense, the list is not a logical “netlist”, but rather a method for checking connectivity and navigating the diagram.

Each item in the pin list is formatted as follows:

For normal devices:*device*-*pin* (*page ref*)

For pseudo-devices:*device type (page ref)*

The elements of this format are as follows:

| | |
|---|---|
| **device** | If the device is named, this will be its name, otherwise it will be a “#” mark plus the type name. |
| **pin** | If the pin has a pin number, this will be the pin number, otherwise a “#” and the pin name. |
| **page ref** | This will be a reference to the page number and grid position, formatted as specified in the Design Preferences box. |

***For more information on page references, see “Automatic***

***Display of Page References” on page 219.***

### Selecting a Signal

A single signal is selected by clicking anywhere along the signal line. This selects only the part of the signal directly attached to the clicked line. Double-clicking the signal selects all parts of the signal on the given page including logical connections by name or bus.

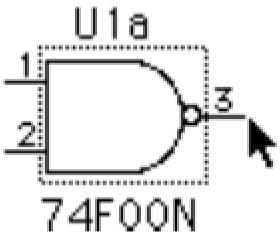
### Selecting a Pin

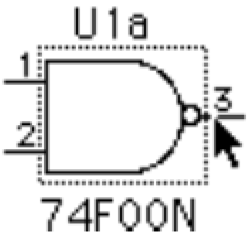
A pin is selected by clicking on the pin line close to the device.

**NOTE: Since an unconnected device pin is both a pin and a signal, you determine whether you get the pin or signal pop-up menu as follows:**

clicking on the pin in the last 1/4 of the pin length away from the device will display the signal menu.

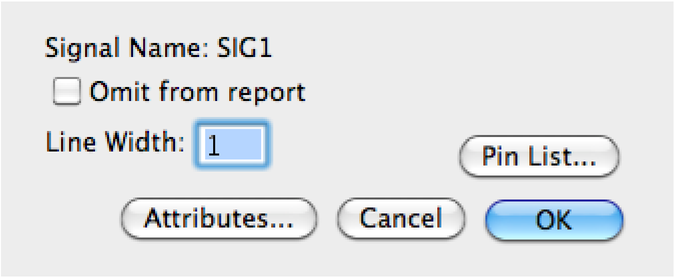
clicking on the pin close to the device symbol will display the pin menu.





### Getting and Setting Signal Information

Selecting the Get Info command with a signal selected causes the following box to be displayed:



Signal Name: SIG1

☐ Omit from report

Line Width: 1

This box can also be displayed by holding the control key pressed while clicking and holding on a signal line. In the pop-up menu, select the Signal Info command.

Here are the options presented in this box:

| | |
|---|---|
| **Omit From Report** | This switch controls whether the selected signal is included in any netlist output. See “Signal Reporting Options” on page 372 for more information. |
| **Line Width** | The number typed into this box determines the displayed width of the signal. Any number > 1 displays the signal that number of times wider than the normal value. The maximum value is 255. |
| **Pin List** | This button displays a list of the real device and pseudo-device pins attached to this signal. Double-clicking on any item in this list will display the selected pin. More information on this box is given in “Checking Signal Interconnection” on page 96. |
| **Attributes** | This button displays the general attribute data entry box for the selected signal. More information on the functions available in this box are given in “Entering and Editing Attribute Data - Basic Procedure” on page 175. |

## Naming Signals

Names may contain any letters, numbers or special characters that you can type on the keyboard, but are restricted in length to at most 15 characters. The name associated with an object can be placed anywhere on the diagram and will be removed if the object is removed.

### What Signal Names are Used For

The signal name is referenced by the following DesignWorks functions:

The signal name is used in report output, such as netlists and error checking reports.

The signal can be located by name using the Find tool.

Signals can be logically interconnected by name.

## Invisible Signal Names

DesignWorks allows you to assign and edit signal names in a circuit without making them visible on the diagram. This can be used to assign names that cannot be conveniently placed on the diagram, or that are needed for report generation purposes only. Signals with invisible names are not connected by name, except for invisible names created by a Signal Connector device, as described above.

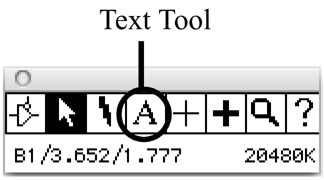
### Connecting Signals by Name

Signal names can be used to make logical connections between wires that are not visually connected on the schematic. Like-named signal traces on a single page are logically connected for simulation and netlisting purposes. Whenever a signal name is added or changed, the circuit is checked for a change in connectivity. If the name is now the same as another signal on this page, the two signals are merged into one. If this signal segment was previously connected by name to others and the name is changed, then the logical connection is broken. Whenever a name change causes two signals to be connected, both parts will flash on the screen to confirm the connection.

***More information on connecting across pages, using power and ground connectors, using busses, and other signal connection issues are covered in detail in Chapter 10—Making Signal Connections on page 207. In particular, you may wish to refer to "Signal Connectivity Rules" on page 232.***

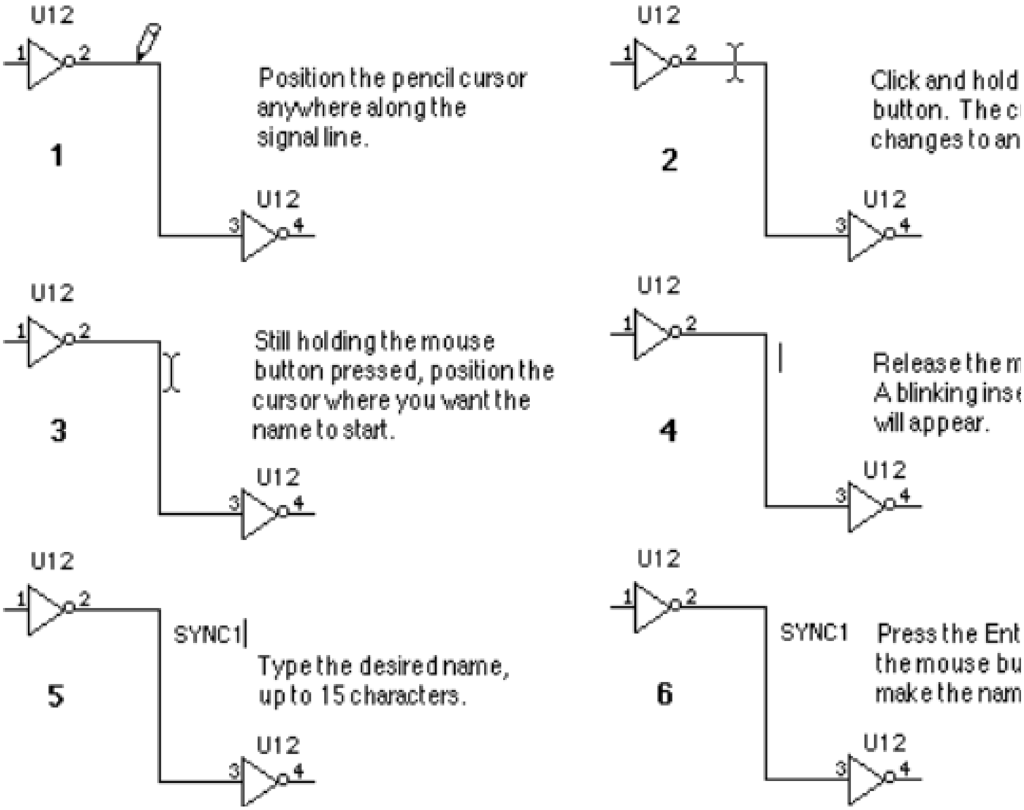
### Adding a Signal Name

Enter "Text" mode either by selecting the Text command in the Edit menu, or by clicking on the text icon in the tool palette



Note that once "Text" is selected, the cursor changes to a pencil icon.

Press and hold the mouse button with the tip of the pencil positioned anywhere along a signal line except within 5 screen dots of a device. As long as you hold the mouse button down an I-beam cursor will track the mouse movements. The signal name text will start at the position where you release the button. Type the desired name and press Enter or click the mouse button anywhere. This is illustrated below.



Position the pencil cursor anywhere along the signal line.

Click and hold
button. The c
changes to an

Still holding the mouse button pressed, position the cursor where you want the name to start.

Release the m
A blinking ins
will appear.

Type the desired name, up to 15 characters.

Press the Ent
the mouse bu
make the nam

## Adding an Invisible Name

An invisible name for a signal can be created by either:

1) Holding the control key while clicking on the signal, then selecting the Name command from the pop-up menu.

OR

2) Selecting the desired signal, then selecting the Get Info command in the Options menu (Command-I), then clicking on the Attributes button in this box, then selecting the Name field in the attributes box.

In either case, if the name is already visible on the diagram, changing it here will change all displayed occurrences of it.

**IMPORTANT: If two signals are given the same invisible name, they will not make a logical connection in the DesignWorks schematic, since they must be visible for the connection to be made. However, some PCB or simulation systems consider any two signals with the same name to be connected. In order to minimize the chance of creating an accidental short between two signals with the same name, the DesignWorks auto-naming feature will always assign a new, unique name to any signal that has an invisible name that is copied, duplicated or otherwise edited on the schematic.**

### Making an Invisible Name Visible

An invisible name can be made visible by simply clicking the Text cursor anywhere on the signal or device. When the mouse button is released, the name will be positioned at that point, as shown in the general naming instructions above. Alternatively, you can use the Name command in the signal pop-up menu to change the Visible status of the name.

### Removing a Name

A signal name can be removed by using the Zap cursor. If the signal has been named in multiple locations then Zap removes the name only at the location zapped, but the signal still retains the logical name. Alternatively, you can hold the Control key while clicking on the name text on the schematic and use one of these commands in the attribute pop-up menu:

The Hide command can be used to hide the name without changing it.

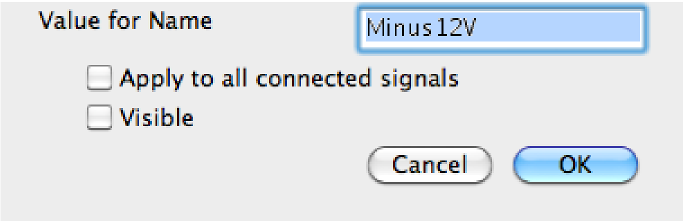
The Clear command can be used to remove the name completely.

**IMPORTANT: See "Adding an Invisible Name" on page 101 for an important note about invisible signal names and auto-naming.**

## Editing a Name

The name can be changed by simply clicking the Name cursor on the signal name and editing using the keyboard as desired. Alternatively, a name can be edited by using the Name command in the signal pop-up menu.

This command displays a name edit box for the selected signal. The following box is displayed offering you several options on how the name changed is to be applied:



Value for Name: Minus12V

☐ Apply to all connected signals

☐ Visible

### Apply to All Connected Signals

This option allows you to choose whether the name change applies only to the selected signal segment (i.e. thereby breaking its connection with other like-named signals), or to all interconnected signal segments.

### Visible

This option allows you to choose whether the name entered should be displayed on the schematic. If the name was already visible and you uncheck this box, it will be removed from the schematic. In this case, the name will still be associated with the signal as an invisible attribute. If the name was not previously visible and you check this box, it will be displayed somewhere adjacent to one of the signal line segments.

## Moving a Signal Name

A signal name can be moved by selecting the Point (normal) cursor, clicking and holding the mouse button on the name, and dragging it to the desired new position. Holding the Option key depressed while moving a name (or any attribute value) disables the grid snap, allowing you to micro-position the name for alignment with other graphic items on the schematic.

**NOTE: The "Name" is actually an attribute field, so all attribute editing techniques can be used on it. In particular, you can**

**hold the Control key while clicking on the name and the attribute pop-up menu will appear, giving you a variety of editing commands, described in “Other Ways of Viewing and Editing Attributes” on page 183.**

### Multiple Naming of Signals

A signal name can appear in up to 100 positions along the length of a signal line. To add a new position, simply use the normal naming procedure given in the section on signal naming i.e.:

Select Name mode

Click and drag anywhere along the signal line

With the mouse button pressed, move to the desired position for the name

Release the mouse button

A new copy of the signal’s name will appear at this point followed by a flashing cursor. To accept the name, simply click the mouse button once or press the Enter key. If you edit any occurrence of a name along a signal segment, all other occurrences will be updated to reflect the new name.

Any occurrence of a signal name can be removed using the Zap cursor. If you remove the last visible name from a signal segment then the logical connectivity to other like-named signals is removed.

## Using the Auto-Naming Features

Two features are available to simplify the naming of groups of related signals, devices and pins. These features are activated by holding down the Command, Option or Shift keys while selecting the signal to be named with the Text cursor.

Auto-alignment - If the Command key is held down while the signal is selected, the text insertion point will be positioned horizontally aligned with the last signal name that was entered. The vertical position is determined by the vertical position of the line that was clicked on. This feature works only with signal names, not with devices or pin numbers.

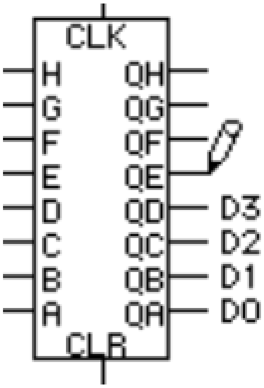
Auto name generation - If the Option key is held down while a signal, device or pin is selected, a new name is generated automatically for this item. The new name will be the same as the last one entered, except that the numeric part of the name will have been incremented. If the previously-entered name did not have a numeric part, then a "1" digit will be appended to it. If the Shift key is pressed at the same time, the number will be decremented instead of incremented.

### Sequential Naming

The above two features can be used in combination to perform easy naming of sequential signals. The normal symbol standard in DesignWorks is to position the highest numbers at the top, so you can either:

number the topmost line in the group (e.g. D7) using the normal naming technique, described above, then hold Command and Option while clicking on successive lower-numbered lines, or

number the bottom-most line in the group (e.g. D0) using the normal naming technique, described above, then hold Command, Option and Shift while clicking on successive higher-numbered lines.



Note that when you select each successive line, the new name appears, then it is necessary to click again or press Enter to make the name permanent.

## Pin Numbering and Information Entry

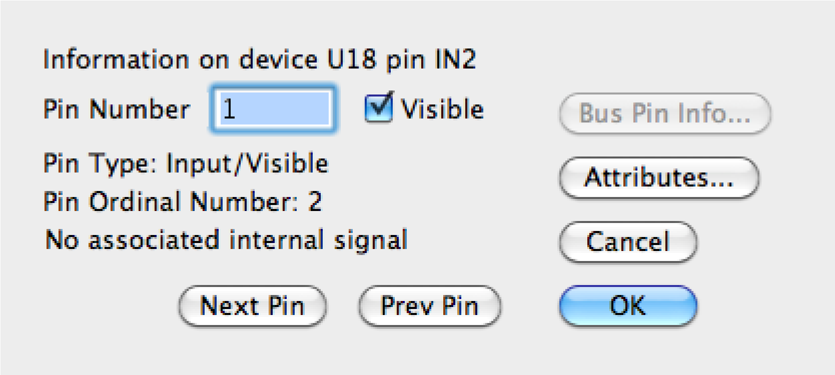
Pins are circuit objects that provide a link between devices and signals. A pin cannot exist independently of a device, but it can have a variety of information associated with it that can be set separately for each pin. This information includes the visible pin number on the schematic and any number of attribute fields.

## Getting and Setting Pin Information

You can display a general pin information box by either selecting the pin and choosing the Get Info command in the Options menu, or, holding the Control key pressed while clicking on the pin, then selecting the Pin Info command in the pin pop-up menu.

***See "Selecting a Pin" on page 98 for the technique used to select a pin.***

In either case, this box will be displayed:



Information on device U18 pin IN2

Pin Number: 1

☑ Visible

Pin Type: Input/Visible

Pin Ordinal Number: 2

No associated internal signal

The following information and options are available:

| | |
|---|---|
| **Pin Number** | This is the physical pin number corresponding to this device pin. This can be empty if desired. See "Uses of Pin Numbers" on page 107 for more information. |
| **Visible** | This check box determines whether the pin number is displayed on the schematic. For some devices, such as discrete components, it may be desirable to have a pin number associated with the pin for netlist purposes without displaying it on the diagram. |
| **Pin Type** | This information item gives the function and visible status of the pin. |
| **Pin Ordinal Number** | This number is the pins ordinal position in the device's pin list, i.e. as viewed in the DevEditor. This can be important in some netlist formats where pin order is critical. See "Reordering Pins in the Pin List" on page 308 for more information. |

| | |
|---|---|
| **Associated Internal Signal** | For sub-circuit devices, this item shows the name of the signal attached to the associated port connector in the internal circuit. For other devices, this will be "None". See "Making Signal Connections Across Hierarchy Levels" on page 262 for more information. |
| **Attributes** | This button displays the general attribute data entry box for the selected pin. See "Entering and Editing Attribute Data - Basic Procedure" on page 175. |
| **Bus Pin Info** | This option will be enabled if the selected pin is a bus pin. This will display the Bus Pin Info box which provides a number of information and editing operations for working with bus pins. See "Using Bus Pins" on page 214 for more information. |
| **Prev Pin/ Next Pin** | These buttons allow you to move to the next or previous pin (by ordinal number) on the same device, without having to return to the schematic and select the pin. |

## Pin Numbering

Pin numbers may contain 0 to 4 characters (not necessarily just numeric) and are always positioned adjacent to the associated pin.

### Uses of Pin Numbers

Pin numbers are used only for labeling purposes and have no particular connectivity significance to DesignWorks. Pin numbers are not automatically checked for duplicates or other invalid usage although you can use the ErrorScript tool to run such error checks manually. Pin numbers placed on a diagram can be used in creating a netlist, and will appear when the circuit is printed. If a pin is unnumbered, it will appear in a netlist with a "?" unless the $AUTONUMBER option is used in the report script.

***For information on extracting pin numbers in netlists, see the ReadMe file provided with the design kit you are using and refer to the $AUTONUMBER and $PINNUM script commands in the DesignWorks Script Language Reference (separate manual on disk).***

### Default Pin Numbers

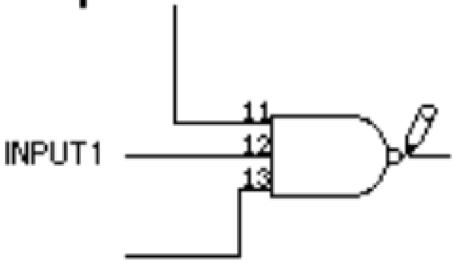
A device type may have default pin numbers associated with it which will appear when the device is first created. These pin numbers are not permanent and can be edited or removed by the techniques discussed in this section. Devices in a library can have default pin numbers assigned to them using the DevEditor tool.

***See "Setting the Pin Number" on page 307 for information on setting default pin numbers while editing a symbol.***

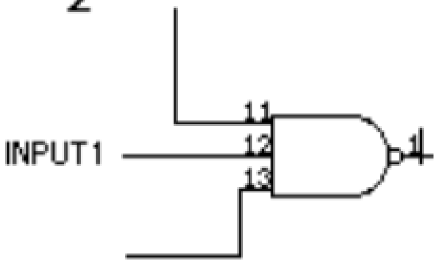
### Editing Pin Numbers On the Schematic

If the mouse button is pressed with the tip of the pencil positioned on a signal line within 5 screen dots of a device, a blinking insertion bar will appear immediately where the signal joins the device. You cannot set the text position for pin numbers. Type the desired 1 to 4 character number and press Enter, or click the mouse button anywhere, to make the number permanent. See the sequential numbering option discussed below under "Auto-naming Features". Pin numbers cannot be repositioned.

1



2



### Quickly Applying Sequential Pin Numbers

If the Option key is held down while a pin is clicked on with the pencil cursor, a new pin number is generated automatically for that pin. The new number will be the same as the last one entered, except that the numeric part of the string will have been incremented. If the previously-entered item did not have a numeric part, then a "1" digit will be appended to it. If the Shift key is pressed at the same time, the number will be decremented instead of incremented.

### Editing Pin Numbers Using Get Info

To edit pin numbers using the Get Info dialog box:

1) Click on the device in question to select it.

2) Click on the Pin Info box. This will display the pin information for the first pin.

3) Edit the pin number as desired.
4) Click the Next Pin button to see the next pin in the list.

### Editing Pin Numbers Using the Browser

To edit pin numbers using the Browser tool:

1) Select the Browser command in the Tools menu.
2) Select the Pins object type.
3) Select the Number item in the Pseudo pop-up menu.
4) Select the Edit item in the Mode pop-up menu.

Pin numbers can now be edited in the “Number” column of the spreadsheet.

### Setting Pin Number Text Characteristics

The text font used for displaying pin numbers is setting by selecting the Design Preferences command in the Drawing menu, then clicking the Pin Text... button. This displays the text specification box, allowing you to select a font family, face and point size. Any changes made to this text style will be applied to all pin numbers displayed throughout the design. There is no ability to select different fonts for individual devices or pins. The text changes are applied when the OK button in the Design Preferences box is clicked.

### Allowing Rotated Pin Numbers

You can optionally show pin numbers on all north- and south-facing pins rotated 90° to run along the length of the pin. This is set by selecting the Design Preferences command in the Drawing menu, then checking the Allow Rotated Pin Numbers item. If this item is unchecked, all pin numbers will be displayed horizontally adjacent to the pin.

## Using Text and Picture Objects

Text and picture objects are used only to enhance the graphical appearance of a schematic diagram. They have no logical significance in the design.

Text objects are not associated with any particular device or signal on the screen and should not be used to set a name or attributes for devices or signals. The text in these boxes is not accessible in net or component lists. Use the naming and attribute features to attach text to devices and signals.

## Creating a Text Block

If you click the pencil cursor on the diagram not near a device or signal line a blinking cursor will appear at that point and you will be able to type any desired text on the diagram. The Return key can be used to enter multiple lines in a single text block. Text entry is terminated by the Enter key.

## Editing a Text Block

If you click the pencil cursor inside an existing text item the insertion point will be positioned at the click point. You can then use normal Macintosh text editing techniques to modify the text. Note that text on the clipboard can be pasted into an existing text box using the Command-V key equivalent for the Paste function. The Paste menu command will cause the current text entry to be terminated and a new text box to be created. Similarly, the Command key equivalents for Cut (X) and Copy (C) can also be used while editing a text box.

Text boxes can be Zapped, Duplicated, Cut, Copied, Pasted and dragged just like any other item on the screen. See the descriptions of these commands for more information.

## Using Text Variables

The text facility allows special entries called variables to be placed in text boxes. When the text is displayed or printed, the appropriate system quantity or attribute field value is substituted for the variable name. Any variable name that is not recognized will be displayed verbatim. This facility is intended to assist in creating title blocks and similar additions to the schematic with information such as the date, page number and title, revision level, engineer's name, etc. Using variables will allow this information to be entered in one place so that changes are reflected automatically on all schematic pages.

### System Variables

System variables start with a "$" and are enumerated here:

| | |
|---|---|
| **$DATENOW** | This will be replaced by the system date written mm/dd/yy. This is the current date as maintained by the Macintosh clock. |
| **$TIMENOW** | This will be replaced by the system time written hh:mm. This is the current time as maintained by the Macintosh clock. |
| **$DATECREATED** | This will be replaced by the date the file was created, as maintained by the Macintosh file system. |

## Background and Border Objects

Either text or picture blocks can be set to be "border" or "background" objects. These two options will normally be used together, but can be enabled separately if desired. These two options are enabled by selecting the object in question and using the Get Info command.

### Border Objects

If a text or picture is marked as a "border" item, then it is considered to be part of the sheet border information for the page it is on. This has the following effects:

> if a new page is added to the circuit using the Pages command, this item will automatically be copied to the same position on the new page.
>
> if the page containing this item is used as the source for an Import Sheet Info operation, this item will be copied to the destination.
>
> if a Center in Page command is used, this object will not be repositioned since it is assumed to be part of the border.

### Background Objects

Text or picture items marked as "background" objects cannot be selected, modified or deleted using normal circuit editing techniques. This is used to prevent custom sheet backgrounds or border graphics from interfering with circuit editing operations. This is normally used in combination with the "border" status discussed above.

To select a background object, hold the Command and Option keys pressed while clicking on it. The Get Info command can then be used to disable the background status.

| | |
|---|---|
| **$TIMECREATED** | This will be replaced by the time the file was created, as maintained by the Macintosh file system. |
| **$DATEMODIFIED** | This will be replaced by the date the file was last modified, as maintained by the Macintosh file system. |
| **$TIMEMODIFIED** | This will be replaced by the time the file was last modified, as maintained by the Macintosh file system. |
| **$PAGENUM** | This will be replaced by the number of the page (within the current circuit level) the text is on. |
| **$NUMPAGES** | This will be replaced by the total number of pages in the circuit level. |
| **$PAGETITLE** | This will be replaced by the title of the page the text is on, as entered using the Pages command. |
| **$PRINTPAGENUMBER** | This will be replaced by the page number in the printed sequence within the current Print Design command. This is used to number pages within a hierarchical design. NOTE: This is only valid for printed output. When it is drawn on the screen, it will show the same value as $PAGENUM. |
| **$PRINTNUMPAGES** | This will be replaced by the number of pages that will be printed by the current Print command. This is used to number pages within a hierarchical design. NOTE: This is only valid for printed output. When it is drawn on the screen, it will show the same value as $NUMPAGES. |
| **$FILENAME** | This will be replaced by the name of the design file. |
| **$CIRCUITNAME** | This will be replaced by the name of the circuit being printed. In the topmost circuit of a hierarchical design (or in any flat design), this is the same as $FILENAME. In a sub-circuit, this will be the hierarchical name of the circuit. |

## Attribute Variables

Attribute variables start with a "&" mark and are used to refer to fields stored in the attributes for the design. These can be used to place information at multiple points on a diagram which will be updated automatically when the design attributes are changed. For example, if the design attribute field Revision was defined for the design with the following contents:

```
2.1A Mar 18, 2011
```

then the variable &Revision would appear as “2.1A Mar 18, 2011" on the diagram.

***See Chapter 9—Attributes on page 173.***

### Editing Text with Variables

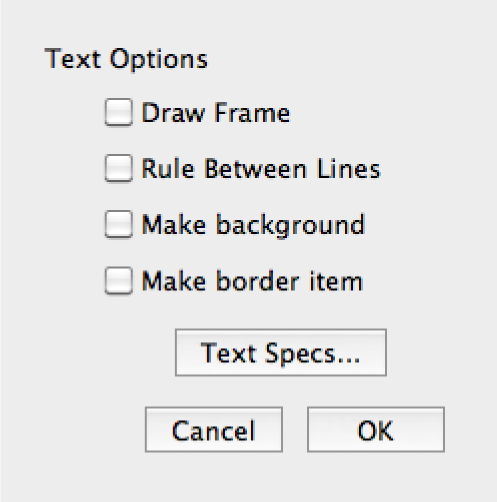
Text items on the schematic will normally be displayed with variables replaced by their values. When a text item is selected or clicked on with the Name cursor it is redisplayed in its raw format with the variable names shown without interpretation. This allows the items to be edited with the text in its actual stored position.

### Text Frame Size with Variables

The framing rectangle for a text item is calculated after the variable substitution is done. This may cause the item to be highlighted or deleted incorrectly if the variable values are shorter than the names. These errors are not serious or permanent and will disappear when the screen is updated. The text box can be expanded as necessary by adding blanks at the end of any one line in the item.

## Text Style and Display Options

To set text display options and text style, select the text block by clicking on it with the Point cursor, then select the Get Info command in the Options menu. This will display the following box:



This table summarizes the options available.

| | |
|---|---|
| **Draw Frame** | Turning this switch on causes a frame to be drawn around the text item on the schematic. |
| **Rule Between Lines** | Turning this switch on causes a line to be drawn after each row of characters. |
| **Make Background** | Turning this switch on makes the selected text block a background item, i.e. it cannot be affected by normal editing operations. In order to select it for editing, hold the Command and Option keys pressed while clicking on it. |
| **Make Border** | Turning this switch on marks this item as a border object. This means that it will be treated as part of the sheet border. That is, it will be updated by the Import Sheet Info command, copied to new pages as they are created, and unaffected by the Center in Page command. See “Setting Sheet Sizes and Borders” on page 350 for more information. |
| **Text Specs** | Clicking this button displays the standard text style dialog. Any changes made in text style affect only the selected item, but they also become the default for future text blocks. |

***See more information on using text notations in title blocks and sheet borders in “Setting Sheet Sizes and Borders” on page 350.***

### Selecting a Picture or Text Object

A single, non-background picture or text item is selected by clicking the mouse button with the pointer positioned anywhere inside the item.

### Selecting a Background Object

To select a picture or text item marked as “background” you must hold the Command and Option keys pressed while clicking on it.

## Creating a Picture

Pictures cannot be created from scratch right on the schematic diagram. They can be created either in another application or using the DevEditor tool. To do this:

Enter the DevEditor tool, or other paint, drawing or drafting program to create the graphics object.

Select the desired object.

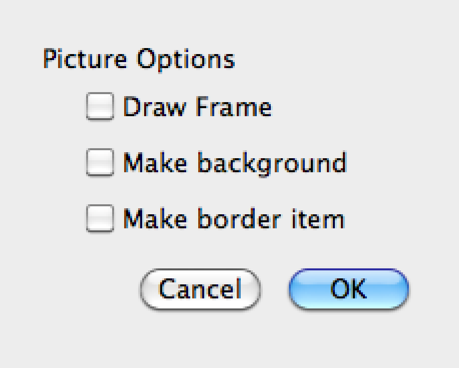
Select the Copy command in the Edit menu. This places a copy of the graphics item on the clipboard.

Select your DesignWorks circuit.

Select the Paste command from the Edit menu. A flickering image of the graphics object will follow the mouse movements until you click once in the desired position.

### Setting Picture Characteristics

If a picture object is selected, the Get Info command displays the following box:



The following table summarizes the options available for pictures.

| | |
|---|---|
| **Draw Frame** | Turning this switch on causes a frame to be drawn around the picture item on the schematic. |
| **Make Background** | Turning this switch on makes the selected picture a background item, i.e. it cannot be affected by normal editing operations. In order to select it for editing, hold the Command and Option keys pressed while clicking on it. |
| **Make Border** | Turning this switch on marks this item as a border object. This means that it will be treated as part of the sheet border. I.e. it will be updated by the Import Sheet Info command, copied to new pages as they are created, and unaffected by the Center in Page command. |

***See "Setting Sheet Sizes and Borders" on page 350 for more information on creating title blocks and sheet borders.***

# Before Starting a Major Design

# 6

This chapter provides information that should be considered before starting any major design. We will take a look at the overall design process, focussing on the schematic diagram. Correct choices made in advance concerning usage of hierarchy mode, attribute format, sheet layout etc. will save time and frustration later.

What uses will the design data be put to?

Are you going to pass data to a PCB design, synthesis, simulation or other design package?

Does your organization have a parts database you need to work from?

How are hard copies to be printed or plotted?

This chapter is not intended as a specific reference for any DesignWorks features. It focuses on your desired end-product and how best to make use of this package to achieve that goal. Its primary aim is to raise issues that you should consider and alternatives that you may wish to test before proceeding.

***This chapter makes general reference to third-party products and specific issues on interfacing with them. Refer to Read Me files provided with the design kits included with DesignWorks for more information on specific systems.***

## The Golden Rule - Try a small design first!!!

Regardless of how simple your design process may seem, a small test of the entire flow will pay large dividends in reduced future headaches. If your design process involves moving netlist or other data files between software systems, file compatibility is an important issue. Refer to the design checklist given in the next section and create a small design that will exercise the areas of each system that may cause problems.

# Design Process Checklist

Every design process has its own unique elements:

Company standards for printed documents,
Software and hardware compatibility,
Device symbol requirements

and so on. Even if all you want to do is draw a small schematic and print it on your local laser printer, there are issues of fonts, sheet sizes, logos, etc. that will affect your final result.

In the following sections we present a checklist of issues that may affect your design process. Many of the items in the list will probably not apply to your situation. Make your own list of issues that need a second look and produce a small test design to check them out before proceeding with a large design.

The points given here are intended primarily to prompt you to look at issues in your design process, rather than to provide any specific information on DesignWorks features. In many cases, references are given to other areas in this manual where further information can be found.

## Schematic Creation

In this section, we'll list the issues that affect the creation and printing of the schematic diagram itself.

### Attribute Usage

The DesignWorks *attribute* facilities provide a powerful mechanism for storing arbitrary text information with a schematic. This information can be shown on the printed sheet and/or passed to external systems in netlists and reports. Defining a clear list of the attribute data required in your design will help reduce the effort required to get the right text displayed on the printed sheets and the right data passed out to your PCB layout or simulation package.

***Complete details on the attribute facilities in DesignWorks are given in Chapter 9—Attributes on page 173.***

Here are some ideas to consider regarding using attributes in your designs:

Define fields for design before creating libraries—The set of attribute fields used in symbol definitions in libraries is independent of any particular design. While this creates a great deal of flexibility, it can also lead to problems if you have variations in attribute field naming and value format. It pays to give some thought in advance to how you will name fields (e.g. upper case first letter, underscores between words, etc.) and how you will format part names, component values, etc.

Applying attributes on the schematic—DesignWorks has a number of options for fonts, text rotation and positioning of attribute fields. You may wish to create and print a small design first to verify that things will appear the way you want in the final hardcopy.

Back annotating attributes—Some applications require bringing text information back into the schematic from other systems. The most common application is PCB package names, but other simulation and layout data may be candidates as well. You should determine if this data is available in some machine-readable format and if it can be automatically read back in by DesignWorks.

Design attributes for Designer, Revision, etc., PCB Info, Simulation Info—You may wish to create a number of design attribute fields that can be displayed on each sheet using the text variable facility described in "Using Text Variables" on page 110.

Device attributes—part names, package codes, in-house part number, etc., simulation info, PCB info, gate packaging—Device attribute have a wide range of applications in interfacing to external systems. You should confirm what part information is needed by the target system before proceeding with creating a design or its symbol libraries.

Pin attributes: gate packaging, simulation info—Some layout and simulation systems require specific names or other information on device pins.

Signal attributes—Signal attributes are less commonly used, but may be needed to store PCB track width, cabling information or other data. Especially in cabling applications that require a sequence of cable and connector information to be stored with a net, you should confirm that the data can be stored and output in the format you need.

## Schematic Portability

In some applications, the portability of schematic data between systems may be an issue. DesignWorks runs on both MacOS and Microsoft Windows-based machines and there are some issues of text font usage and background graphics that may affect the movement of data between machines.

In addition, some companies or projects may require that schematic data be stored in some standard form for archival purposes when a project is finished.

## Power and Ground Nets

There are a variety of methods that are commonly used to show power and ground connections on a schematic and to represent them in netlists and reports. For each design, you should decide what power and ground nets are needed and whether you wish to show them explicitly as lines on the schematic or use attributes or signal connector symbols. More information on this topic is provided in “Power and Ground Connections” on page 223.

## Symbol Libraries

Device symbols are an important resource in your design creation process. Whether you primarily use the symbols provided with DesignWorks, or you create special libraries for your own use, the completeness and accuracy of this data has a major effect on your design flow. Library files generally outlast any one design and are used for many years across many projects, so they become an important asset for your company.

Here are some symbol-related issues that you may want to consider:

Symbol standards—Does your company or customer require any specific symbol standards?

Attributes—Do you have a clearly defined set of attribute fields that you want to include in each symbol?.

Pin ordering for reports—Some netlist formats, notably SPICE, require that pins appear in a very specific order. This is determined by the order in which you place the pins when creating a symbol.

Library organization—Library organization becomes an issue especially if you are working with a team of designers. Are project libraries shared in a central location? Who is in charge of maintenance? Does each designer have their own custom libraries?

## Hierarchical Design

Hierarchy allows you to present a system design at different levels of abstraction, from a simple block diagram at the top to specific physical details at the bottom. While it may seem that this would always be beneficial, in practice, hierarchy does not lend itself to all applications. DesignWorks offers three hierarchy modes to suit a variety of applications. Here are some points to consider before settling on which to use.

Hierarchical design is a powerful technique, but does add some complexity to the design process. If you are one of the following:

a new user of schematic capture tools

working with small-to-medium designs (say, less than 200 ICs)

interfacing to a third-party Printed Circuit Board layout package

...then we recommend sticking with "Flat" mode and not using hierarchical blocks in your designs. Once you have gained some experience with creating schematics in DesignWorks and moving design data between systems, you can review the hierarchy issues discussed in this manual “Choosing a Hierarchy Mode” on page 238.

If you are working as part of a team where another individual will be responsible for the manufacturing of the final product, and another person again for the long-term maintenance of the design, then hierarchy may not be appropriate. Hierarchy is a good concept for thinking about and presenting system design concepts, but is not good for tracing physical connections from part to part. Many companies do not allow the use of hierarchical design for PCB-based projects.

Once you have decided to use hierarchy, you need to decide which of the DesignWorks hierarchy modes to use. whether to use. Pure hierarchy mode is intended for large designs that:

lend themselves to a very structured hierarchical definition.

do not require device instance data.

do not require flat netlist output.

Unless you are sure your design meets these requirements, do not use Pure mode. Pure mode is not recommended for printed circuit board-level designs.

Physical hierarchy mode is intended for designs that require data to be associated with each physical device instance. This will be the case for most PCB designs and any design using the DesignWorks Simulator option.

***See Chapter 11—Hierarchical Design on page 235 for more information.***

## Printing and Plotting

The size and format of the schematic sheets you use will likely be determined by your organization's drawing standards and by what form of hardcopy device you have available. It is best to do some test sheets in your chosen format before proceeding with a large design. Reformatting a design to a different sheet size is an unpleasant task.

### Sheet Layout

You will need to choose a sheet size that suits your project and the available hardcopy output devices. DesignWorks allows each page of a circuit to be different, if necessary, so you may wish to have a different layout for a title page, an "unused gates" page, or other parts of the design. If you want a company logo in your title block, it is best to test this on your output device and make sure it prints or plots correctly.

### Border Size and Scaling

Most laser or dot-matrix printers have a number of options for print area size and scaling. For this reason, it is best to either:

Use the "Use Page Setup" option and default borders in the Custom Sheet Info command. This way, the schematic drawing area will automatically adjust to changes in printer page setup.

OR

Choose a format and scale factor that looks good on your printer and create a fixed sheet template for it. Use the "Fit to single sheet" option in the Custom Sheet Info box to force the printer to scale the output to the available printing area.

***For more information on sheet settings and scaling, see "Fitting the Diagram to the Available Paper" on page 62.***

## Reports

DesignWorks has a powerful report generation and scripting capability that can be used to produce text reports in a variety of formats.

Bills of Materials—Does your company or purchasing department have a standard for machine-readable or hardcopy reports?

Error Reports—DesignWorks includes error checking scripts for some common types of errors. Are there other errors that could be caught before sending your design out to the PCB house?

Exporting to Spreadsheets and Databases—DesignWorks has the ability to create tab-delimited text data formats that can be read directly most spreadsheet or database applications. This may allow you to simplify the entry of data for project costing and documentation.

## Interfaces to Other Systems

No design process is complete without considering the final destination of your schematic data. In many cases, the simulation or layout package that you are interfacing to will impose rules on device and symbol naming and attribute formats.

Netlist format—The single most important issue in exporting data to a third-party system is the format of netlist required. DesignWorks includes a large collection of netlist generating scripts and also allows you to create your own. Is the one you need supported? Can you get the format information needed to create your own? What information does the format require for each device or signal in a design? As with other issues presented here, it is best to test a small design first and make sure these issues are resolved before investing a large effort in a design project.

Attributes required—Most PCB or simulation packages will require some extra information to be entered into your libraries or schematics, such as package codes, simulation parameters, etc. In some cases it may be possible to use a translation table in a netlist script to generate package codes for a target system from the ones provided in the DesignWorks libraries. Alternatively, you can consider back-annotating the desired values into the design from an external database. In any case, you should confirm that your chosen method is going to work before proceeding.

Naming restrictions—Most external packages have some restrictions on name length or special character usage in names. This is especially an issue if you are using hierarchical design to create the schematic, since this can result in long names being automatically generated. In addition, you should be clear on the case sensitivity of names in the target system. Are signals "CLOCK" and "Clock" going to be treated as the same thing?

Back annotation—If your design process requires back annotating information from a PCB layout system into the schematic, make sure the format you have available is supported in DesignWorks.

## Documentation

Most projects have some requirement for written documentation in addition to the schematics. If you plan to include any graphics extracted from schematics, you should verify that you can cut and paste these successfully from the schematic into your word processor.

DesignWorks can also generate report outputs in a variety of formats that may be incorporated into written documentation.

# Device Packaging and Naming

## 7

This chapter provides information on the automatic gate packaging and naming features for devices in DesignWorks. Device name assignment is an important issue because the device name frequently has special significance when data is exported to another design system. For example, if you are using a SPICE-based simulator, the first letter of the name determines its model type. If you are working with a PCB-layout package, the name is used to hold the package assignment.

The auto-packaging feature is intended for PCB-related applications, where multiple logical symbols on a schematic may be grouped together in a single physical device package on a board. DesignWorks handles this by allowing you to store information in attributes that indicate the number and type of logical devices in a package.

Auto-naming is a simpler feature that ensures that every symbol that is placed in a circuit has a unique name assigned. Auto-naming does not take packaging information into account.

This chapter also discusses device date stamping. Date stamping is a mechanism that automatically marks devices with a time value when they are created. This can be used in conjunction with some PCB systems for forward- and back-annotation purposes.

## Packaging vs. Auto-Name Assignment

The Packaging and Auto-Name Assignment features can both be used to assign names to devices. Both of these features are controlled using the Device Naming and Packaging Options menu command in the Naming and Packaging sub-menu of the Options menu. These are the key distinguishing features of these two options:

> Packaging assigns a name, unit and pin numbers to each device based on physical packaging considerations using information stored in attributes with the symbol. Note that packaging does not take any layout considerations into account, it simply assigns the next

available package to each symbol as it is placed.

Auto-naming assigns a name to each device symbol in a circuit without looking at any attribute information except the default name prefix. This ensures that each device symbol has a unique name.

Packaging operates across an entire design whereas auto-naming applies separately to each circuit level in a hierarchical design. That is, in a hierarchical design, there can be two or more devices with the same name, e.g. “R23”. See “Using Hierarchical Names” on page 249 for more information on using hierarchical names to distinguish objects in a hierarchical design.

In flat designs, the only difference between auto-packaging and auto-naming is in the use of the packaging information during name assignment. If the device libraries you are using have no packaging information, these two mechanisms will have exactly the same effect.

## Choosing Which Options are Appropriate for your Design

Whether you choose to enable the auto-packaging or auto-naming options depends upon your particular application, the hierarchy mode being used and on your personal work style. For more information on choosing a hierarchy mode, see “Choosing a Hierarchy Mode” on page 238. Once that issue is settled, here are some issues that will affect the naming and packaging options that you use:

It almost always makes sense to have either or both auto-naming and auto-packaging turned on. Even if you will be changing the names later based on some other layout considerations, using one or the other of these features ensures that every symbol can be easily distinguished on the schematic or in a netlist or report.

If your design is destined for a PCB layout, then it is best to have auto-packaging enabled, for the same reasons given above. In addition, auto-packaging will give you a running idea of how many packages have been used, even if the actual package assignments will be changed later.

If your design is not intended for PCB layout, e.g. a SPICE simulation or an FPGA design, then it is best to use only auto-naming. With auto-packaging, there is a possibility of assigning the same name to multiple devices, which may confuse the target layout system.

In designs using flat hierarchy mode (the default), you can only have one or the other of these options enabled. That is, if auto-packaging is enabled, auto-name assignment will have no effect. This is because both these mechanisms assign values to the Name attribute field in flat designs.

In physical hierarchy mode, both features can be used together, since the name is placed in the Name field, while the package assignment is placed in a separate attribute field, InstName. See “Using the Name and InstName Fields” on page 185 for more information on the use of the InstName field.

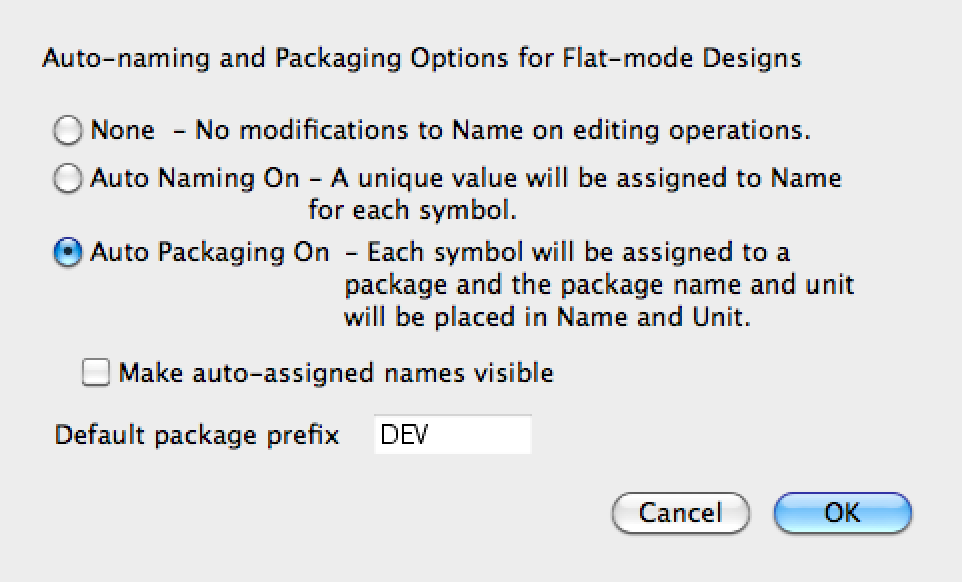
In pure hierarchy mode, packaging is not available since it is not possible to store the physical assignments of devices inside subcircuits that have been used multiple times. Auto-naming is available and is on by default.

## Enabling Naming and Packaging Options

The naming and packaging options that are available to you depend upon which hierarchy mode you are using.

### Naming and Packaging Options in Flat Hierarchy Mode

If the current design is in flat hierachy mode, selecting the Device Naming and Packaging Options menu command will display this box:



Auto-naming and Packaging Options for Flat-mode Designs

○ None – No modifications to Name on editing operations.

○ Auto Naming On – A unique value will be assigned to Name for each symbol.

● Auto Packaging On – Each symbol will be assigned to a package and the package name and unit will be placed in Name and Unit.

☐ Make auto-assigned names visible

Default package prefix: DEV

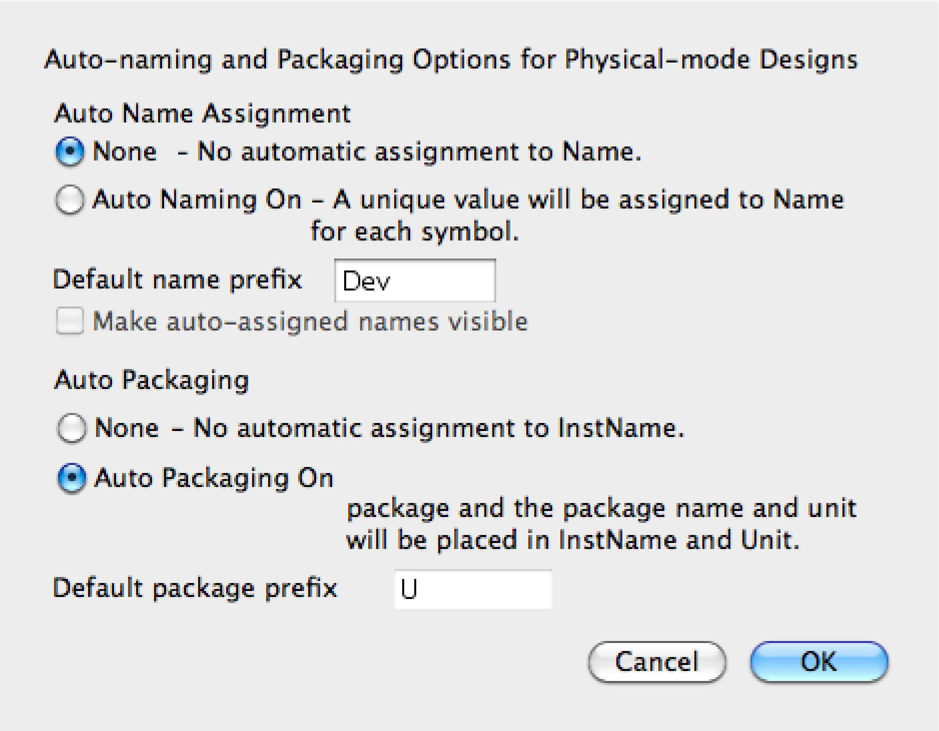
The following options are available:

| | |
|---|---|
| **None** | Both options are off. The Name attribute field will not be modified when a device symbol is placed. |
| **Auto Naming On** | A unique (within the circuit) name will be generated in the Name field when a symbol is placed. |
| **Make auto-assigned names visible** | If this box is checked, the auto-assigned name will be displayed on the schematic next to the symbol. If this box is not checked, the name will be stored with the device, but not displayed on the schematic. |
| **Auto-Packaging On** | A package assignment will be made for each symbol placed on the schematic and the resulting information stored in the Name and Unit attribute fields. The Unit assignment will cause the pin numbers on the symbol to be updated, if that information is provided with the symbol. |
| **Default name/ package prefix** | The text in this box will be the default prefix used to generate a device name, if no prefix is provided in the symbol attributes. See “Setting the Auto-Generated Name Format” on page 136 for more information. (Note that the title on this box changes depending on which option is enabled.) |

## Naming and Packaging Options in Physical Hierarchy Mode

If the current design is in physical hierachy mode, selecting the Device

Naming and Packaging Options menu command will display this box:



Auto-naming and Packaging Options for Physical-mode Designs

Auto Name Assignment

● None - No automatic assignment to Name.

○ Auto Naming On – A unique value will be assigned to Name for each symbol.

Default name prefix: Dev

☐ Make auto-assigned names visible

Auto Packaging

○ None - No automatic assignment to InstName.

● Auto Packaging On

package and the package name and unit will be placed in InstName and Unit.

Default package prefix: U

In physical hierachy mode, the auto-packaging and auto-naming options can be enabled separately. The following naming options are available:

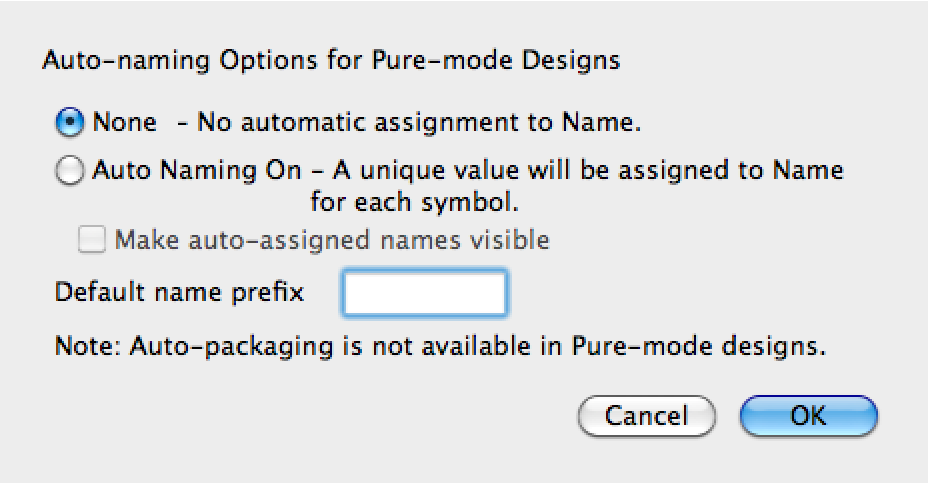
| | |
|---|---|
| **None** | Auto-naming is off. The Name attribute field will not be modified when a device symbol is placed. |
| **Auto Naming On** | A unique (within the circuit) name will be generated in the Name field when a symbol is placed. |
| **Make auto-assigned names visible** | If this box is checked, the auto-assigned name will be displayed on the schematic next to the symbol. If this box is not checked, the name will be stored with the device, but not displayed on the schematic. If auto-packaging is also enabled, you will probably want this to be off so that you don't get a confusing display of two different names. |
| **Default name prefix** | The text in this box will be the default prefix used to generate a device name, if no prefix is provided in the symbol attributes. See "Setting the Auto-Generated Name Format" on page 136 for more information. |

The following packaging options are available:

| | |
|---|---|
| **None** | Auto-naming is off. The InstName attribute field will not be modified when a device symbol is placed. |
| **Auto Packaging On** | A package assignment will be generated in the InstName field when a symbol is placed. |
| **Default package prefix** | The text in this box will be the default prefix used to generate a package name, if no prefix is provided in the symbol attributes. See "Setting the Auto-Generated Name Format" on page 136 for more information. |

## Naming and Packaging Options in Pure Hierarchy Mode

If the current design is in pure hierachy mode, selecting the Device Naming and Packaging Options command will display this box:



Auto-naming Options for Pure-mode Designs

● None - No automatic assignment to Name.

○ Auto Naming On – A unique value will be assigned to Name for each symbol.

☐ Make auto-assigned names visible

Default name prefix

Note: Auto-packaging is not available in Pure-mode designs.

Note that auto-packaging is not available in pure hierarchy mode. The following naming options are available:

| | |
|---|---|
| **None** | Auto-naming is off. The Name attribute field will not be modified when a device symbol is placed. |
| **Auto Naming On** | A unique (within the circuit) name will be generated in the Name field when a symbol is placed. |

| | |
|---|---|
| **Make auto-assigned names visible** | If this box is checked, the auto-assigned name will be displayed on the schematic next to the symbol. If this box is not checked, the name will be stored with the device, but not displayed on the schematic. |
| **Default name prefix** | The text in this box will be the default prefix used to generate a device name, if no prefix is provided in the symbol attributes. See “Setting the Auto-Generated Name Format” on page 136 for more information. |

## Using Device Packaging

DesignWorks provides features for automatic or manual assignment of devices to physical packages. This mechanism is intended for use when creating designs that are destined for production in printed circuit board (PCB) or other discrete package format.

The Packager tool has the following features:

- Automatically generates a unique package assignment for each device placed on the schematic.
- Uses a name prefix stored with the device symbol, or a default value stored with the design. This allows appropriate names to be generated for discrete parts, e.g. "D1", "D2", "D3", etc. for diodes, and "R1", "R2", "R3", etc. for resistors.
- Automatically handles device types having multiple units per package, e.g. gates. Special attributes are stored with the symbol to tell the packager how many units are available and the pin assignments for each.

When Auto Packaging is enabled (i.e. the Auto Packaging On item is checked in the Device Naming and Packaging Options command), the Packager assigns each device to a package as soon as it is placed on the diagram. When a device is deleted, its package assignment becomes available for the next device of the same type that is created. In this mode, package assignments will be in chronological order of placement on the diagram.

It is normally convenient to leave Auto Packaging turned on while initially creating a design. This way, all devices get reasonable initial name assignments. The design can be repackaged later once the physical layout of the design becomes more clear. Repackaging may be done by any of these methods:

Using the Repackage Design command in the Naming and Packaging Options menu. This will assign symbols to packages sequentially based on their position in the schematic.

Manually, by changing the Name (or InstName) and Unit attributes.

Using the BackAnno tool or other automatic back annotation method to read package assignments from a PCB layout system.

## Re-enabling Auto-Packaging After Manual Edits

If Auto-Packaging has been disabled while manual editing was done, then the design's package table will be out of date. It may be desirable to select the Rescan Design command from time to time to recheck the validity of the current package assignments. This command brings the package table up to date based on name and unit assignments currently in the design, without changing any values. In any case, Auto-Packaging cannot be re-enabled until a successful Rescan is completed.

## Auto-Packaging Limitations

The Packager can be confused by some conditions when package assignments are changed manually. Such conditions include:

Two devices given the same packaging assignment.

A name or unit attribute field set to an invalid value.

Packager attributes changed.

In these cases, the Packager will display a warning box showing the error location. After the warning has appeared, the Packager will attempt to continue assigning symbols to packages, although its accuracy cannot be guaranteed.

In order to re-synchronize the Packager with the design, you should take these steps as soon as possible:

Correct the erroneous assignment, or mark the offending device with the "Ignore" option described in “Setting Device Packaging Options” on page 139.

Select the Rescan Design command in the Naming and Packaging sub-menu.

**NOTE: The Packager looks only at the name and unit attribute fields, not at the pin numbers themselves. If you manually change any pin numbers on devices without making corresponding changes to the Unit field, invalid package assignments will result.**

## Bringing the Design's Package Table Up to Date

The Rescan Design command forces the Packager to rebuild the design's package table from the current state of all device attribute fields. This can be used in the following circumstances:

To check for errors or obtain a free unit report after manual packaging changes have been made.

After back annotation or any other external process that may have changed package assignments.

After changing any of the attribute fields that may affect packaging.

If any packaging errors are encountered (e.g. duplicate gate assignment or invalid attribute fields), they will be reported at the end of this process and the design's package table will be considered invalid. No device attribute fields are affected by this command.

## Getting a Report of Unused Gates

The Scripter tool is capable of generating a report of the packages in the design that have not been completely used. All PCB-related design kits supplied with DesignWorks include a script for generating such a report.

***For information on creating your own report format that includes a free unit report, refer to the entry for $UNUSEDUNITS in the DesignWorks Script Language Reference (separate manual on disk).***

**TIP: Some designers like to place a text notation on the diagram that explicitly lists unused elements in the design. This can be done in one operation using a script that combines the $UNUSEDUNITS and $PLACETEXTBLOCK**

**commands. See the DesignWorks Script Language Reference (separate manual on disk).**

## Batch Repackaging the Entire Design

The Repackage Design command in the Options menu allows you to request a complete reassignment of device packages. This command performs the following steps:

Clear the design's package table.

Scan all devices in the design, placing all devices marked as "Lock and Check" or "Lock and Don't Check" in the package table. This ensures that these names will not be used for automatic unit assignment. Devices marked "Ignore" are not placed in the table, meaning that their names could be used in automatic unit assignment. Any errors detected in the package assignment of devices marked "Lock and Check" are reported.

Sort all devices in the design by hierarchy level, then by page, then by horizontal grid position, then by vertical grid position.

In the sorted order defined above, assign a new package name and unit to all devices not marked "Lock" or "Ignore".

For packaging, devices are sorted by page number, then by position on the page. Names are assigned in vertical strips corresponding to the lettered grid references on the page. I.e. all devices in column A are assigned, starting at the top of the page and working down. Next, all devices in column B are assigned, etc.

## Performing Manual Packaging

All aspects of device packaging can be controlled manually, if desired. This can be done on two levels.

### Manual/Automatic Packaging

The higher-level approach takes advantage of the packaging information in the device libraries and the error checking capabilities of the Packager. The following packaging features can be used:

Each library entry contains information about available gate package assignments. The list of available units appears in the Unit sub-menu of the device pop-up menu. Selecting one of these items automatically updates the Unit field and the pin number assignments on the device.

The Browser tool can be used to view and change all name and unit assignments for the design. Changing the Unit value from the Browser will automatically update the pin numbers (assuming the pin numbers are correctly set up in the packaging attribute fields).

The Rescan Design command can be used to check for packaging errors without affecting any existing settings. Any duplicate assignments or invalid unit settings will be announced.

The following cautions should also be noted:

The Packager does not check pin numbers when checking package assignments. It uses only the contents of the Unit attribute field. If you manually change any pin numbers on devices without making corresponding changes to the Unit field, invalid package assignments will result.

It is best to set the Packaging Level to Lock (using the Get Info command) for devices that you wish to have a fixed package assignment, even if not using auto-packaging. This will avoid loss of data if auto-packaging is inadvertently enabled or a Repackage Design is done.

### Fully Manual Packaging

The lower-level approach completely ignores all packaging features. This can be used if unusual pin assignments are needed, or libraries without packaging information are being used. Note the following points when doing manual packaging:

Pin numbers can be edited directly on the diagram or assigned using the Pin Info box or the Browser tool.

If you do not wish to display a unit assignment (e.g. "a", "b", "c", etc.), you can use the Define Attribute Fields command to turn off the Visible by Default option for the Unit field.

If you wish to manually assign a unit value, it is best to create a user-defined attribute field (e.g. "GateUnit") and make all entries in that field. This is necessary to circumvent the automatic updating of pin numbers when the Unit field is changed.

When a netlist or Bill of Materials is created using the Scripter tool, all devices with the same name are normally treated as a single device. This means that they are, in effect, assigned to the same package. The Scripter tool does not check for duplicate pin numbers or name assignments, although it is possible to customize a report format that will assist in locating these problems.

## Setting the Auto-Generated Name Format

An auto-generated device name consists of two parts, the fixed prefix and the numeric suffix. The prefix portion is derived from one of two sources:

The device's prefix attribute field, or if that is empty,

The value set using the Device Naming and Packaging Options menu command.

The numeric suffix is assigned sequentially for each different prefix found in the design. E.g. the packager will assign names "U1", "U2", "U3", etc. to devices with the value "U" in their prefix field, and names "R1", "R2", "R3", etc. for devices having prefix "R".

**NOTE:**

**1) The default prefix field is Name.Prefix. This can be changed by entering the name of any other field in the design's PrefixField field.**

**2) Most library parts provided with DesignWorks that represent logic devices or integrated circuits have <u>no</u> Name.Prefix value. They will therefore use the value of the design's PkgPrefix field, "U" by default. Discrete parts or others that normally have specific standard prefixes are set to the common values.**

**TIP: There are no options in this version of DesignWorks for setting the number origin for name assignment, for example to start at R100 instead of R1. However, you can force this behaviour by creating a separate page in your circuit and placing a number of devices with the names you wish to have unused. If needed, you can mark these devices with the “Omit from Report” option to ensure they do not appear**

**in any netlist output. Alternatively, you can use a Scripter script to assign names in any desired format as a batch process after placing all your parts.**

## Batch Reassigning Device Names

The Reassign Device Names command is a more restricted version of the Repackage Design command and is intended for use when packaging is not being used, or for reassigning names in Physical hierarchy mode.

The Reassign Device Names command assigns a new name to all devices in the current circuit that are either unnamed or have a default name. It is intended as a quick means of tidying up automatically-assigned names when a circuit is created or edited.

**NOTE: If the design is in "Flat" hierarchy mode (the default) then this command will override the gate package assignments made by the Packager. In other hierarchy modes, the Packager stores the package assignments in the "InstName" field, which is unaffected by this command.**

Note the following rules:

Only the current circuit is affected. Other circuit levels higher or lower in the hierarchy are unaffected.

Gate packaging information is ignored, i.e. every device symbol is assigned a unique name regardless of package availability or "locked" status.

The "Unit" attribute field is unaffected.

Only the "Name" field is changed, regardless of hierarchy mode.

Names are assigned in order sorted first down the columns of the references grid, then across the page, then through subsequent pages.

Names are assigned using the device's prefix field (i.e. the one named in the design's "PrefixField" field) or the design's "DevPrefix" field.

Names that have been edited by the user are no longer considered "default" names and will not be changed.

## Setting the Name Prefix for a Symbol

The name prefix is set by filling in the appropriate attribute field while editing the part in the DevEditor. For example, for a resistor, you would place the value “R” in the Name.Prefix field using the DevEditor’s Set Part Attributes command.

***See “Setting Part and Pin Attributes” on page 295 for more information on using this command.***

## Specifying that a Device Should be Unnamed When Placed

In some cases it may be desirable to have a device remain unnamed when it is placed on a schematic. An example might be a symbol that is used as a place holder for some global schematic information or design parameters, a test point, I/O pad, etc. but does not represent a real device.

The keyword "$NONAME" can be placed in the Name.Prefix field in a device symbol to indicate that no name should be assigned when the symbol is placed.

## Selecting an Alternate Prefix Field

The prefix used in generating device names is derived from another device field known as the "prefix field". The default prefix field is "Name.Prefix". DesignWorks allows the prefix field to be selected for a given design so that multiple prefixes for different naming conventions can be stored with the same part. For example, the standard Discretes library included with the package has two prefix fields:

> "Name.Prefix" is the normal one used by default for generating names.
>
> "Name.Spice" contains prefix values specific to SPICE-based analog simulators.

In addition, the standard field "Function" contains a short function code for the part that can also be used as a descriptive prefix.

To set the prefix field:

1) Select the Set Design Attributes command in the Options menu.
2) Select the PrefixField item in the field list.
3) Type the name of the desired prefix field in the data box.

**IMPORTANT: The prefix field name must match an existing field name**

**exactly, including case, or it will be ignored.**

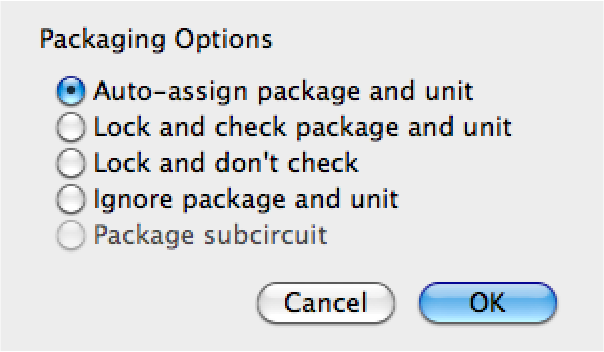
## Setting Device Packaging Options

DesignWorks allows you to mark individual devices for special handling during packaging operations. To set device packaging options, bring up the Device Info box by either one of the following methods:

> Select the device by clicking on it, then select the Get Info command in the Options menu.
>
> OR
>
> Hold the Control key while clicking on the device, then select the Device Info command in the pop-up menu.

Next, click the Packaging Options button in the Device Info box:

This displays the following box:


Packaging Options
● Auto-assign package and unit
○ Lock and check package and unit
○ Lock and don't check
○ Ignore package and unit
○ Package subcircuit
Cancel
OK

These options are described in the following table.

| | |
|---|---|
| **Auto-assign package and unit (the default value)** | This device will get a package name and unit number assigned automatically by the Packager. The assignment can be changed by future repackaging operations if necessary. |

| | |
|---|---|
| **Lock and Check** | This device's package assignment will not by changed, but it is recorded in the package table so no other device will be assigned the same name. The device will be checked for packaging errors. |
| **Lock and Don't Check** | The same as Lock and Check except that the package assignment is not checked for errors. This could be used in special cases where the same name is purposely assigned to two different symbols. |
| **Ignore** | This device is completely ignored by packaging operations. I.e. its name and unit will not be changed, no error checking is done, and some other device might be given the same name. |

The packaging level is actually stored in the PkgLevel attribute field of each device. You can set a default value for a symbol by placing the appropriate text in this field when creating a symbol. See “Setting Packaging Attribute Fields While Creating a Symbol” on page 143 for more information on this process.

## Back Annotation and Packaging

Back annotation refers to any process that reads a list of name, unit and pin numbering changes from an external system and automatically updates the design file. Back annotation may be performed manually or using the DesignWorks Back Annotation tool. Obviously, any such process can potentially bypass the Packager and create invalid package assignments. It is recommended that the Rescan Design command be used after any such process to ensure that the name and unit assignments are still valid.

**WARNING: The Rescan Design command does not guarantee the correctness of your design. Pin numbers are not checked. Check your design carefully after any Back Annotation process.**

## Using Packaging in Hierarchical Designs

### Hierarchy Mode

The usage of attribute fields by the Packager is different between flat and physical hierarchy modes. See the discussion under . Packaging is not available in pure hierarchy mode.

### Name vs. InstName

In hierarchical designs, a device symbol in a sub-circuit can represent multiple physical devices. This happens if the parent device has been instantiated multiple times in the design. For this reason, a single device symbol has two different values that might both be considered a "name" in a hierarchical design. These two values are stored in the following two attribute fields:

The "Name" attribute field is associated with the definition of the sub-circuit and will be the same for all instances of the sub-circuit. For this reason, it can be thought of as the "logical name" or "definition name" of the device and cannot be used to hold the package assignment.

The "InstName" attribute field is associated with each instance of the sub-circuit. It can take on a different value for each physical device it represents and is used to store the package assignment in hierarchical designs. You can choose to display either or both of these names on the schematic.

**IMPORTANT: In Flat mode designs, you *should not* use the InstName field unless you have a very specific reason for doing so. InstName is used only for package assignments in Physical mode designs and will not be correctly output in netlists and reports in Flat mode.**

***See "Using the Name and InstName Fields" on page 185 for more information.***

### Restricting Packaging Depth

In a hierarchical design, a device symbol that represents a physical device may have a sub-circuit for simulation or analysis purposes. Normally, the parent device rather than the contents of this sub-circuit should be packaged. The packaging options in the device Get Info command can be used to control packaging of internal circuits. Standard library devices provided with DesignWorks will default to packaging the parent device even if an internal circuit is added.

**NOTE: The depth restriction on a device is stored in its Restrict attribute field. You can set a default value for this field when creating a device symbol. See "Setting Packaging Attribute Fields While Creating a Symbol" on page 143.**

### Name Assignment Order

In hierarchical designs, all packagable devices in a given circuit level are assigned before the internals of any sub-circuit devices in that circuit. This ensures that a given circuit level contains sequential device names and units.

## Using Device Libraries Without Packaging Information

If the Packager encounters a device without any packaging attribute fields define, it takes the following minimum action:

The device is assumed to have only one unit per package, i.e. each device will be given a unique name.

The default package name prefix for the design will be used to assign a name.

No unit assignment will be displayed on the schematic.

No Unit selection menu will appear.

Default pin numbering defined in the library will be used.

In summary, the device will be named as if it has one unit per package.

# Creating a Symbol with Multi-gate Packaging

A multiple gate package is single physical component that may be represented by a number of separate symbols placed in a schematic.

**NOTE: Even though this mechanism can be applied to types of components other than strictly gates, we will refer to each logic component as a "gate" for this discussion. If this is being used for connectors, for example, then each connector pin (or group of pins) is equivalent to a gate as far as the Packager is concerned.**

All the gates in a package may have the same symbol and attributes or they may be different. If every gate in the package has the same symbol and attributes then the package may be stored in a DesignWorks library as one part. If any of the units have different symbols or attributes then they must have an entry in a library for each different type of gate.

The gate is automatically selected by the packager when a device is placed if Auto-Packaging is enabled. The gate may also be manually selected via the "Unit" sub-menu in the device popup menu.

***There are a number of design attribute fields that also affect packaging operation. See "Setting Design Attribute Fields Used By the Packager" on page 150 for more details.***

## Setting Packaging Attribute Fields While Creating a Symbol

A number of attribute fields are used by the Packager when it assigns a name and unit to a device. If you intend to create your own library symbols that require gate packaging, you will need to be familiar with these fields. If you are only using the libraries provided with DesignWorks, or if you have no requirement for gate packaging, you will not need this information.

### Required Packaging Attributes

This table describes the attribute fields must be correctly set for packaging to operate.

***See "Setting Part and Pin Attributes" on page 295 for spe-***

### Attribute Fields Set By the Packager

The Packager stores the package name and unit assignments in attribute fields associated with each device symbol. The fields that are used depend upon hierarchy mode:

| | |
|---|---|
| **Flat Mode** | The package name is stored in the "Name" field and the gate unit (e.g. "a", "b", "c", etc.) is stored in the "Unit" field. This reflects the common usage that the device name used on the schematic is the name of the package on the final PCB. |
| **Physical Mode** | The package name is stored in the InstName field and the gate unit is stored in the Unit field. The Name field cannot be used because a single device in a sub-circuit may actually represent multiple devices on the physical PCB. Each one of these devices requires a separate package assignment. The "InstName" field can take on a different value for each instance. |
| **Pure Mode** | Packaging is not supported in Pure mode. This is because there is no instance data and therefore no place to store different package assignments for multiple instances of the same device. |

***cific instructions on setting attributes while editing a symbol.***

| Field Name | Set In | Description |
|---|---|---|
| Unit.All | Part | Contains a list of all units in the package, e.g. "a,b,c,d", *even if they are not all represented by this symbol.* Unit names are normally a single letter, but can be any combination of letters and numbers up to 16 characters, e.g. "Tom,Dick,Harry,Fred". |
| Unit.List | Part | Contains a list of all the units in the package that can be represented by this particular symbol, e.g. "a,b". In this case (and in most standard gate devices), all the devices in the package use the same symbol, so Unit.List and Unit.All are the same. |
| VisPin.List | Pin | Contains the list of visible pin numbers, one for each unit in Unit.List. The number of items and order of *VisPin.List* must match the order of *Unit.List* such that the first number in *VisPin.List* corresponds to the number of the pin for the first Unit in the package. |

## Optional Packaging Attribute Fields

This table lists fields that are optional, but that you may wish to set them for most packaging cases. These attributes are normally set in the part definition using the Set Part Attributes and Set Pin Attributes commands in the DevEditor tool.

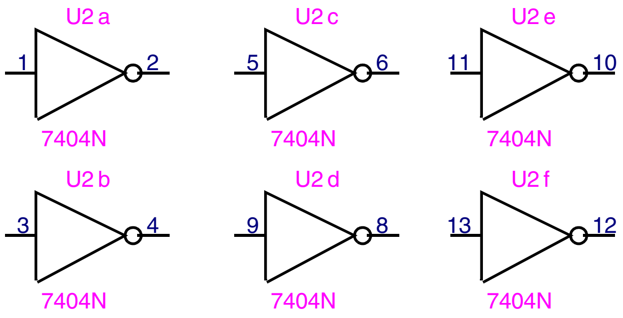
| | |
|---|---|
| **Part** | Contains the part name, e.g. the name that will appear in a bill of materials. The Part field is also used to determine which devices can share a package. This can be any desired name, but must be exactly the same in all symbols that can be combined into a single package. The Packager will never combine two symbols with different Part values into the same package. In devices having several different types of gates in one package, a different symbol is used for each different type of gate, but each must have the same Part value.<br>If there is no value in the Part attribute field, the library type name will be used. |

| | |
|---|---|
| **Name.Prefix** | This attribute field contains the prefix used to generate the device name. If it is empty, then the design's PkgPrefix field is used.<br>NOTE: Name.Prefix is the default field used for this purpose, but this can be overridden. See the PrefixField description in “Setting Design Attribute Fields Used By the Packager” on page 150. |
| **PkgLevel** | This field contains the package level setting for this device instance. If the field is empty, or contains the character "0", Normal mode is assumed. "1" means Lock and Check, "2" means Lock and Don't Check, "3" means ignore. Any other value will be interpreted as Lock and Check. This value is not normally defined in the library entry for the device. The package level is usually set using the buttons in the Device Get Info box, but the attribute field can be set directly if desired. |

***There are a number of other attribute fields that you may wish to set while creating a new device symbol for automatic power and ground connections, etc. See “Setting Part and Pin Attributes” on page 295 for more information on setting attributes in a symbol.***

## Creating Symbol - Multiple Gates - Same Symbol - Example

A Hex Inverter (e.g. 7404) has six symbols in its package. Each of these symbols is identical and has identical attributes, so they can be represented by one entry for a 7404 in the library.



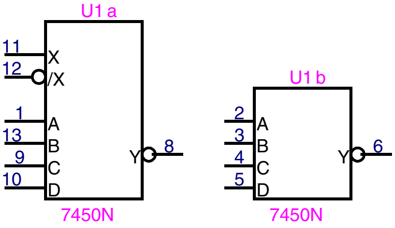
In the above example the packaging attributes are set as follows:

| **Field** | **Value** |
|---|---|
| Unit.List | a,b,c,d,e,f |
| Unit.All | a,b,c,d,e,f |
| VisPin.List on the first pin | 1,3,4,9,11,13 |
| VisPin.List on the second pin | 2,4,6,8,10,12 |

## Multiple Gates - Different Symbols - Example

A 7450 Dual 2-Wide 2-Input AND-OR-INVERT Gate has two units. Each unit has a different symbol and thus two entries in the library, a “7450(N)A.a" and a "7450(N)A.b".

If a package has more than one entry in a library, we annotate this by appending the first unit number of the package that shares the same symbol. If all units are the same then the library name does not include the unit.



The packaging of the And/Or/Invert 7450 is accomplished via these attribute fields:

In the "a" unit:

| **Field** | **Value** |
|---|---|
| Unit.List | a |
| Unit.All | a,b |
| VisPin.List on the first pin* | 11 |
| VisPin.List on the second pin* | 12 |
| Part | 7450(N)A |
| etc.... | |

In the "b" unit:

| **Field** | **Value** |
|---|---|
| Unit.List | b |
| Unit.All | a,b |
| VisPin.List on the first pin* | 2 |
| VisPin.List on the second pin* | 3 |
| Part | 7450(N)A |
| etc.... | |

**NOTE: Since there is only one of each type of unit, it is not strictly necessary to set any value in the VisPin.List field. The pin numbers can be placed on the symbol using the DevEditor in the usual way.**

## Creating a Symbol for a Discrete SIP Package - Example

A resistor "SIP" package is unusual because it has a common pin shared by all the symbols that are combined into a package. In this example, we will create a 9 pin device containing 8 resistors all with a common pin on one side. Pin 1 will be the common pin and pins 2 through 9 will be the 8 resistors.

To create this we will use the *DevEditor* tool and edit the RES symbol provided in the Discretes library (being careful not to save it back under the same name!).

From the DevEditor's DevEdit menu we select "Part Attributes…", This brings up the attributes dialog. Scroll to the "Unit.List" attribute and set its value to "a,b,c,d,e,f,g,h".

**NOTE: The quotation marks '"' are shown for clarity only and are not entered in the value field.**

Next select "Unit.All" and make it the same as Unit.List.

Exit the Part Attributes dialog and then select "Pin Attributes…" from the DevEdit menu. The attributes for pin A should be visible in the dialog. Select the "VisPin.List" attribute and set its value to "2,3,4,5,6,7,8,9".

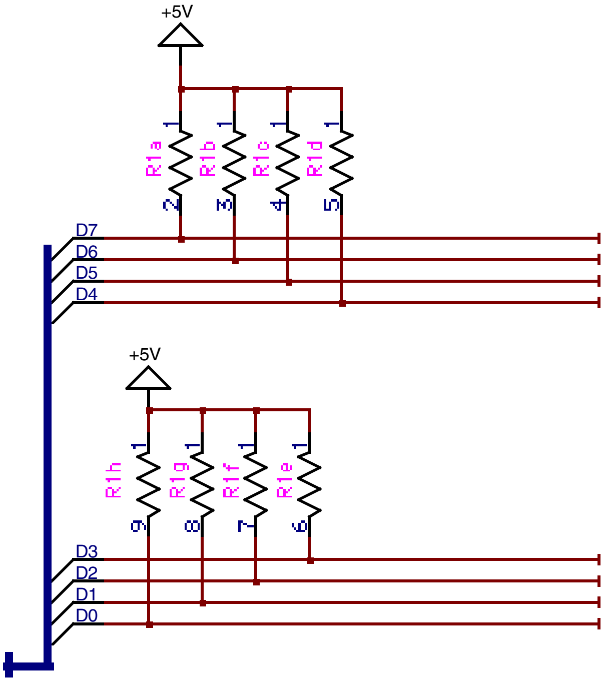
Now press the Next Pin" button at the bottom of the pin attributes The dialog should show the attributes for pin B. Set the VisPin.List attribute to "1,1,1,1,1,1,1,1".

Giving pin B all the same pin numbers for each unit will give the other side of the SIP for each resistor a pin number of 1 so that when individual resistors are placed around a schematic the report generator will merge all the pins together.

**NOTE: It is not strictly necessary to place the value "1,1,1,1,1,1,1,1" in the VisPin.List attribute for pin B. Since there is only one possible value for this pin number, you can also just place the pin number "1" on pin B on the symbol in the usual way and leave VisPin.List empty for this pin.**

Exit the Pin Attributes dialog and use "Save As…" to save the new part to one of your custom libraries. We don't want to change the original RES definition.

Now the new SIP type may be placed around a schematic with each resistor automatically packaged by the packager. It is still up to you to make sure that the common pin 1 is always connected to the same signal.



To change the above example into a resistor DIP package without a common pin, you only need change the VisPin.List for pin A to "1,2,3,4,5,6,7,8" and VisPin.List for pin B to "16,15,14,13,12,11,10,9".

## Specifying PCB Package Type Information

***See "Selecting the Part and Package Type" on page 91 for more information on specifying the package code for use by an external PCB package using the Package attribute field.***

**NOTE: The package code entered in the Package attribute field is used only to pass information to an external PCB layout package. It is not used internally by DesignWorks for gate packaging or any other operation.**

## Setting Design Attribute Fields Used By the Packager

The following information is used by the Packager in determining what units are available in each package and pin assignments belong with each unit. Devices with only one unit per package do not need any special attribute fields.

| | |
|---|---|
| **PkgPrefix** | This design attribute field contains the default package name prefix used if no prefix field appears in the device. |
| **PrefixField** | This design attribute field contains the name of the field to use for a device name prefix. The default is Name.Prefix, but this can be overridden to use specialized prefix values. |

## Packager Error Codes

A number of error messages may be displayed when an attempt is made to package a device with incorrect package attribute data. These are summarized in the following table.

| **Error Message** | **Meaning** |
|---|---|
| An error occurred trying to get a new package name...code 1 | The package name generated by this operation was too long. I.e. prefix is too long or package numbers are getting very large. |
| An error occurred trying to get a new package name...code 2 | Some package names in Unit.List did not appear in Unit.All. |
| An error occurred trying to get a new package name...code 3 | Unit.List was empty when Unit.All was non-empty, i.e. the list of allowable gates was empty. |
| An error occurred trying to get a new package name...code 4 | Internal error - shouldn't occur. |
| An error occurred trying to get a new package name...code 5 | Internal error - memory allocation problem. |
| Error in package attributes or internal packager error | Unit.List or Unit.All are empty, prefix too long, package number too large, or memory error. Occurs during Rescan Design. |

## Device Token Values

Every time a device is created in a DesignWorks circuit, it is assigned an integer value known as its "token". The token number stays with the device for its lifetime and numbers are not re-used. This ensures that a given device can always be recognized despite duplicate names or name changes. The token is used for a number of internal operations in DesignWorks, but can also be seen by the user in the following circumstances:

The token number is displayed in the Device Info box.

The token number can be written out in netlists or bills of materials whenever a guaranteed-unique identifier is needed.

Note the following characteristics of tokens:

Tokens are assigned independently for each circuit in a hierarchical design and are thus only unique within a circuit, not across the entire design.

Each logical symbol on the diagram (including pseudo-devices) has its own token. In a netlist, several symbols may be combined into a single package, so there is not necessarily a one-to-one correspondence between tokens and physical packages.

## Using Packaging with Connector Symbols

Schematic symbols that represent connectors can generally be handled like any other device. However, for large connectors, you may wish to take advantage of gate packaging to allow you to use a number of smaller symbols instead of one large one.

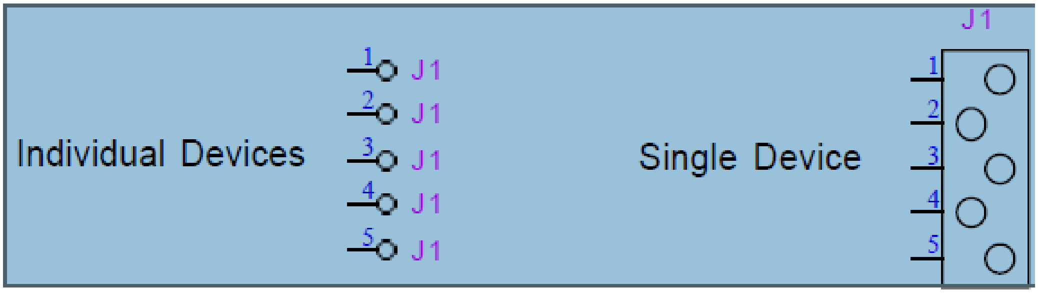
Like any other large component, connectors can be handled in one of these ways:

A special symbol can be created for the connector with the appropriate number of pins and pin numbering specified for each pin. This can be done using the .DevEditor tool to create a device symbol using your own picture.

Each connector pin can be created as a separate, 1-pin device using the symbols from the Connectors library or a custom symbol. This option is preferable only if you need to spread the connector pins over different parts of the diagram. In this case, each "device" must be given the name of the connector and the pin number associated with that pin. We are relying on the fact that the report generator will normally merge all devices with the same name into a single component entry. Attributes (if any) are taken from the first device encountered with the given name.

Connector pins can be grouped with any number of pins in each group. This grouping may be done functionally, e.g. all data pins on one symbol, or simply by number of pins, e.g. 8 pins on each symbol. In the first case, the connector can be treated like a multi-gate package with different symbols being combined into a single package. See “Multiple Gates - Different Symbols - Example” on page 146 for information on creating a symbol for this type of usage. If you are using fixed-size groups, the same symbol can be re-used for each group, with only the pin numbers changing. See “Creating Symbol - Multiple Gates - Same Symbol - Example” on page 145 for an example of creating a symbol with the appropriate packaging attributes for this method.

Following is an example of these methods:



**NOTE: When the single-pin devices are used, every device must carry exactly the same name, although the names can be invisible if desired.**

**TIP: You can use the ConnMaker tool to quickly generate a connector symbol of arbitrary size. See “Auto-creating**

**Connector Symbols" on page 316.**

## Handling Discrete Components

From a packaging and pin number point of view, discrete components such as capacitors, transistors, etc. can be handled just like any other device, although you may want to note the following special considerations.

### Pin Numbering on Discrete Components

Pin numbers are not normally placed on discrete component pins on a diagram. If pin numbers are omitted from a device, DesignWorks will normally put a question mark in the netlist item for that device. Two methods are available to provide pin numbers for netlisting purposes:

> To provide automatic numbering of discrete devices pins, the Scripter provides the $AUTONUMBER option, which can be specified in the form file. This option causes any device with less than or equal to the given number of pins to be numbered automatically if no pin numbers are present on the diagram. In the standard form file, this value is set to 3 to cover the common discrete components.

**WARNING: This option assumes that the pin number order of the discrete components is not significant. If a specific order is important, then you should specify pin numbers explicitly and make them invisible.**

> Pin numbers can be assigned but left invisible. This is done using the Get Info command for either the pin or the device.

## Device Date Stamping

Date stamping is a mechanism that automatically marks devices with a time value when they are created. This can be used in conjunction with some PCB systems for forward- and back-annotation purposes.

The date stamp is stored using the Macintosh's internal integer date format, that is, an unsigned integer representing the number of seconds since January 1, 1904. When a device is created or undergoes any major editing operation, the current time value is converted to a decimal string and assigned to the DateStamp.Dev attribute field. For PCB packages that simply require a unique identifier for devices, this can be used directly with the knowledge that it can be sorted in time order. To create a more human-readable date value, the Scripter has date conversion functions available.

***See more information on date conversions in report scripts, see the section Date and Time References in the DesignWorks Script Language Reference (separate manual on disk).***

## Disabling Date Stamping

The date stamping process incurs a small overhead in processing time and memory space. If you are not using this feature and wish to disable all internal processing related to it, you can place the following line in the DesignWorks Setup file:

```
NODATESTAMP;
```

This will take effect the next time you start the DesignWorks program.

# Tools for Searching, Browsing and Error Checking

# 8

## Introduction

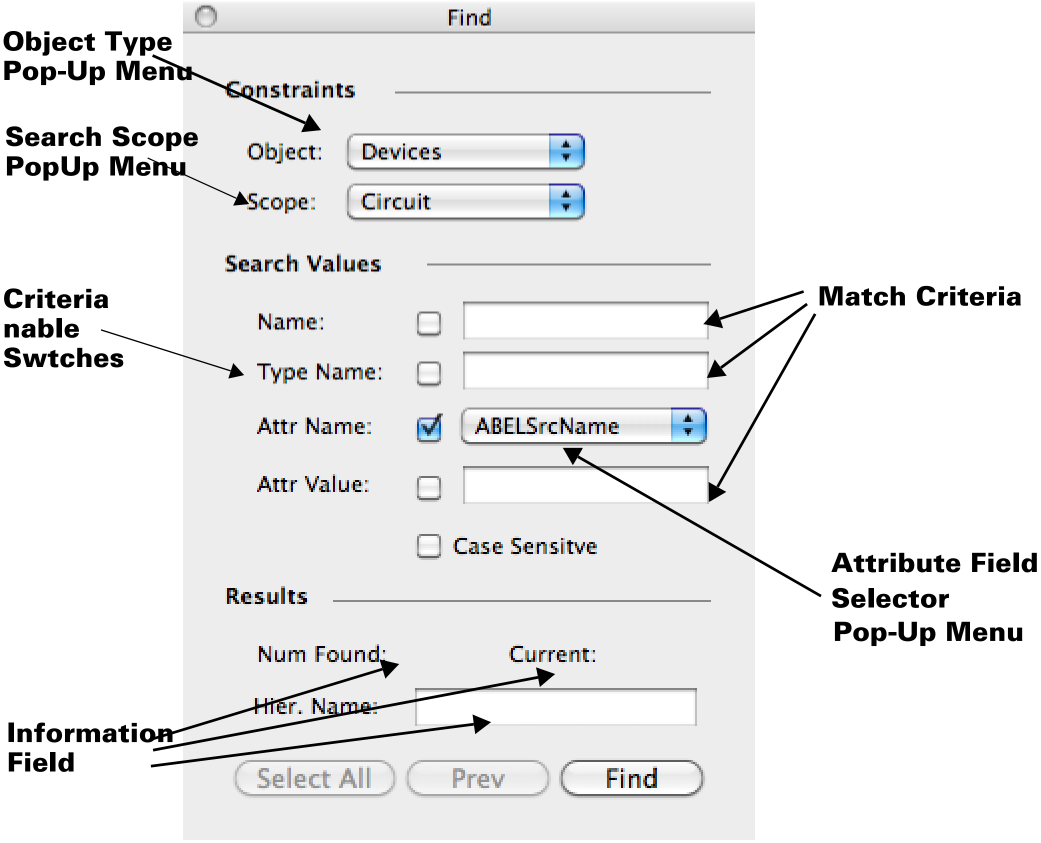
The purpose of this chapter is to describe the usage of a number of tools that can be used to locate and update circuit objects in a design. These include:

- The Find tool is used to locate devices, signals or pins in a schematic by name or attribute values. Once the objects are found they may be sequentially selected and viewed or selected as a group to be operated on with some other tool. For example, you might locate all capacitors with a certain Value setting, then use the Browser to display those selected items and enter a modified value.
- The ErrorScript tool is intended specifically to perform application-specific error checks on designs and to help you locate objects with potential errors. For example, in a PCB design, ErrorScript will look for devices or signals with invalid names or package codes, pins with invalid pin numbers, etc. As its name implies, ErrorScript is script based, that is, it calls the Scripter to run a script file and then uses the output of the script to locate the objects that the script describes. For this reason, it has considerably more flexibility than just as an error locating tool. Any processing that can be done in a script can be used to locate objects. This chapter describes how to use ErrorScript with the predefined scripts provided and also gives details on customizing your own error location scripts.
- The Browser is a tool used to view, edit and navigate schematic data in spreadsheet-style format. Rows of the spreadsheet display objects in the schematic, either devices, signals, pins or nets. A cell of the spreadsheet displays an attribute value for the object in that row and the attribute field name in that column.
- The remainder of the chapter is divided into three sections covering these tools in more detail.

# Using the Find Tool

## Starting Find

To invoke Find, select its name in the Tools menu.  This opens the find palette window:



Object: Devices
Scope: Circuit
Name: ☐
Type Name: ☐
Attr Name: ☑ ABELSrcName
Attr Value: ☐
☐ Case Sensitve
Num Found:
Current:
Hier. Name:

The following sections provide details on the usage of the various controls in this window.

## Search Scope

You can control the types of circuit objects the Find tool will search for and the scope of the search using these controls:

**Object Type Pop-up Menu**  This is a pop-up menu that displays the name of the type of object to be found.  When clicked on, the pop-up menu allows you to select one of the following choices Devices, Signals, Pins, Selected Devices, Selected Signals, or Selected Pins.

**Search Scope Pop-up Menu** This is a pop-up menu that displays the name of the scope to search for objects. When clicked on, the pop-up menu allows you to select one of the following choices.

| | |
|---|---|
| **Design** | All unlocked circuits in the design are searched. |
| **Circuit** | Only the current circuit is searched. |
| **Here/Down** | All unlocked circuits from the current circuit and down in the hierarchy are searched. |
| **Page** | Only the current page is searched. |

## Search Criteria

### Criteria Enable Switches

These checkboxes indicate if their corresponding criteria will be used when trying to find a match. The check box will be automatically selected if a value is entered in the corresponding field.

### Match Criteria

Text entered into these fields is compared to the values in the objects examined to see if they match. The following "wildcard" characters may be used:

| | |
|---|---|
| ***** | Any string of zero or more characters |
| **?** | Exactly one character |
| **#** | Any string of one or more decimal digits |

**TIP: You can search for unnamed devices or signals by clicking on the enable switch next to Name text box, leaving the box empty. A similar method can be used with any attribute field, that is, just select the desired field and check the enable box next to the field value without actually entering a value.**

### Attribute Field Selector Pop-up Menu

This pop-up menu displays the name of the attribute field to be matched.

When clicked on, the pop-up menu allows you to select one of the attribute fields valid for the type of object being searched for.

## Information Fields

These fields may not be edited but instead just display results.  The top field displays the total number of objects found that matched the criteria. The next field indicates which of the found objects is currently selected. The bottom field displays the objects hierarchical name, i.e.. the object's name prefixed by its position in the hierarchy.

### Find Controls

There are three buttons. The first "Select Found Objs" selects all of the objects found. The second button "Prev" moves backwards in the list of found objects and selects the object before the current one. The last button "Find" / "Next" has two operations. Initially, when the button is labeled "Find", it is used to initiate the search. Once the objects have been found then its label changes to "Next" and when pressed it advances the selection to the next found object. You can return to the "Find" by changing any of the search criteria or by making a structural change to the circuit.

## Using the Browser Tool

### Starting the Browser

To invoke the Browser select it from the Tools menu. This opens a spreadsheet window. By default the Browser will initially display all devices in the current circuit. If any objects are selected in the schematic, the Browser will display only selected devices or signals. Selected devices have priority over selected signals.

### Updating the Browser Window

The Browser will update any time a change occurs in the schematic that effects the data displayed in the spreadsheet, or when its scope is set to Circuit and the current circuit changes.

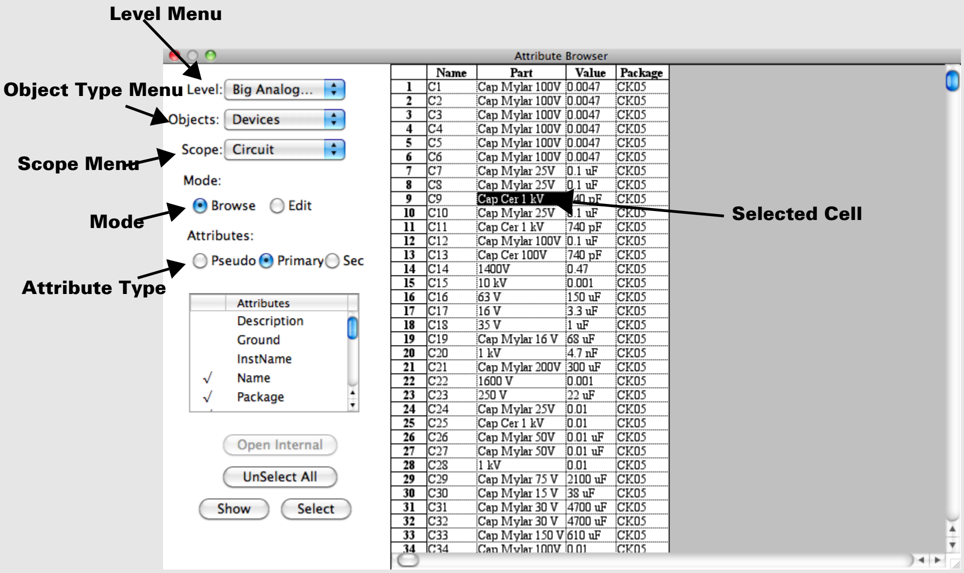
**NOTE: The update is delayed until the mouse is in the Browser's window and the window is at the front.**

### Selecting the Type of Objects Displayed

The object menu pop-up allows you to see and change the type of object being displayed. There are eight different types of objects that may be displayed Devices, Signals, Pins, Nets, and selected Devices, Signals, Pins, and Nets.

If the selected form of an object is chosen to be displayed, e.g. Selected Devices, then only objects of the same type, i.e. Devices, which are selected in the schematic will be displayed. Another example, if Selected

# The Browser Window



Level: Big Analog...
Objects: Devices
Scope: Circuit
Mode: ● Browse ○ Edit
Attributes: ○ Pseudo ● Primary ○ Sec

| | Attributes |
|---|---|
| | Description |
| | Ground |
| | InstName |
| ✓ | Name |
| ✓ | Package |

| | Name | Part | Value | Package |
|---|---|---|---|---|
| 1 | C1 | Cap Mylar 100V | 0.0047 | CK05 |
| 2 | C2 | Cap Mylar 100V | 0.0047 | CK05 |
| 3 | C3 | Cap Mylar 100V | 0.0047 | CK05 |
| 4 | C4 | Cap Mylar 100V | 0.0047 | CK05 |
| 5 | C5 | Cap Mylar 100V | 0.0047 | CK05 |
| 6 | C6 | Cap Mylar 100V | 0.0047 | CK05 |
| 7 | C7 | Cap Mylar 25V | 0.1 uF | CK05 |
| 8 | C8 | Cap Mylar 25V | 0.1 uF | CK05 |
| 9 | C9 | Cap Cer 1 kV | 40 pF | CK05 |
| 10 | C10 | Cap Mylar 25V | 0.1 uF | CK05 |
| 11 | C11 | Cap Cer 1 kV | 740 pF | CK05 |
| 12 | C12 | Cap Mylar 100V | 0.1 uF | CK05 |
| 13 | C13 | Cap Cer 100V | 740 pF | CK05 |
| 14 | C14 | 1400V | 0.47 | CK05 |
| 15 | C15 | 10 kV | 0.001 | CK05 |
| 16 | C16 | 63 V | 150 uF | CK05 |
| 17 | C17 | 16 V | 3.3 uF | CK05 |
| 18 | C18 | 35 V | 1 uF | CK05 |
| 19 | C19 | Cap Mylar 16 V | 68 uF | CK05 |
| 20 | C20 | 1 kV | 4.7 nF | CK05 |
| 21 | C21 | Cap Mylar 200V | 300 uF | CK05 |
| 22 | C22 | 1600 V | 0.001 | CK05 |
| 23 | C23 | 250 V | 22 uF | CK05 |
| 24 | C24 | Cap Mylar 25V | 0.01 | CK05 |
| 25 | C25 | Cap Cer 1 kV | 0.01 | CK05 |
| 26 | C26 | Cap Mylar 50V | 0.01 uF | CK05 |
| 27 | C27 | Cap Mylar 50V | 0.01 uF | CK05 |
| 28 | C28 | 1 kV | 0.01 | CK05 |
| 29 | C29 | Cap Mylar 75 V | 2100 uF | CK05 |
| 30 | C30 | Cap Mylar 15 V | 38 uF | CK05 |
| 31 | C31 | Cap Mylar 30 V | 4700 uF | CK05 |
| 32 | C32 | Cap Mylar 30 V | 4700 uF | CK05 |
| 33 | C33 | Cap Mylar 150 V | 610 uF | CK05 |
| 34 | C34 | Cap Mylar 100V | 0.01 | CK05 |

| | |
|---|---|
| **Level Menu** | A pop-up menu that displays the name of the current circuit level in a design being viewed. When selected, the pop-up menu shows the circuit hierarchy upwards from the current circuit. |
| **Pseudo Attribute Menu** | When selected the pop-up menu shows attributes of the object that are not true attributes, e.g. Grid Position. |
| **Primary Attribute Menu** | When selected the pop-up menu shows attribute names that may be displayed for the selected object type and that are distinguished as primary in the attribute definition dialog. |
| **Secondary Attribute Menu** | When selected the pop-up menu shows the attribute names that may be displayed for the selected object type and that have <u>not</u> been distinguished as primary in the attribute definition dialog. |
| **Selected Cell** | The selected cells are operated on by the Open, Select, and Show buttons, as well as the command pop-up menu. |
| **Row Label** | Generally it just contains a row index, but in Net mode it contains the name of the net associated with the row. |
| **Mode Menu** | A pop-up menu which when not selected displays the current mode. When selected it allows you to choose between Browse and Edit mode. |
| **Scope Menu** | A pop-up menu which when not selected displays where the Browser will look for information. When selected it allows a user to choose between Design, Circuit, Here Down, and Page. |
| **Object Type Menu** | A pop-up menu which when not selected displays what type of object the Browser is displaying. When selected it allows a user to choose between Devices, Signals, Pins, Nets, Selected Devices, Selected Signals, Selected Pins, and Selected Nets. |

Signals is chosen in the object menu pop-up then only signals that are selected in the schematic will be listed in the spreadsheet.

Clicking on the object menu pop-up or making a selection from it will cause a new search to be performed to find the requested objects. The attribute fields displayed the last time this object type was selected will again be displayed.

## Determining Where to Search for Objects

When looking for objects, the Browser will limit its search scope by using the selection made in the scope menu pop-up. There are four different scopes:

| | |
|---|---|
| **Design** | In this scope the entire design hierarchy is search for objects of the correct type. This scope has a side-effect. Some of the objects attribute values displayed may not be editable because of definition/instance conflict resolution. The uneditable objects and/or fields will be displayed with a gray background. |
| **Circuit** | In this scope the search is limited to the pages of the circuit with the most active window. |
| **Here Down** | In this scope the search for objects is in the circuit with the active most window and in all circuits below it in the hierarchy. Some of the objects and/or fields may not be editable for the same reasons as explained in Design scope. |
| **Page** | This scope only looks for objects in the circuit with the most active window and in addition only looks at the page associated with that window. |

## Displaying Attributes

For all objects other then Nets and Selected Nets, attributes may be displayed. Attributes are divided into three groups:

| | |
|---|---|
| **Pseudo Attributes** | These are characteristics about the object that the schematic knows but are not true attributes. These attributes may not be edited. |
| **Primary Attributes** | These are attribute that have been selected in the Define Attribute Fields box in the Schematic Tool and marked as being primary. This is done to allow easier access to the more common attributes. |
| **Secondary Attributes** | These are all the attributes that are not marked as primary (see above). |

The attribute field names displayed in each pop-up menu will change as the type of object selected for display is changed. Only attributes that apply to a given object type are displayed. For example, the attribute field "Part" is associated with Devices but will not be associated with Signals.

In addition, some attribute names in the Primary and Secondary attribute menus will be underlined. An attribute name that is underlined indicates that the attribute value is stored with the instance of the object and not with its definition.

***See "Defining a New Attribute Field" on page 192 for more information.***

If an attribute name is selected in one of the pop-up menus then it will be added as a new column at the end of the spreadsheet. In addition, the attribute name in the menu will have a check mark (Ð) beside it. Selecting a checked attribute name removes the associated attribute from the spreadsheet.

## Changing Attribute Values

### Selecting and Editing Cells

There are two modes of operation:

| | |
|---|---|
| **Browse** | This mode allows you to look at the objects and their attributes in the spreadsheet but not to edit the text of a cell. |
| **Edit** | This mode allows all the same operations as Browse, but in addition allows cells to be edited. |

A cell may be selected by clicking in its bounds. A range of cells may be selected by clicking in another cell with the shift key down. The selected cells will be inverted. If the Browser is in edit mode and the cell is editable (no gray background) then the last cell clicked in will have a flashing edit cursor in it.

Modifying the value of a cell will cause the corresponding attribute value of the object in the schematic to change.

**WARNING: The schematic data is updated immediately and there is no undo.**

### Using the Command Pop-up Menu

Pressing the mouse button with the Control key down, while over the spreadsheet, will cause a pop-up menu to appear for all objects except Nets. This pop-up allows cells to be Cut, Copied, Pasted, and Cleared. There are four additional menu items:

| | |
|---|---|
| **Fill Down** | The Fill Down command takes the selection rectangle and copies the values in the cells at the top of the selection to all the selected cells below them. |
| **Fill Right** | The Fill Right command takes the selection rectangle and copies the values in the cells at the left of the selection to all the selected cells to the right of them. |
| **Make Visible** | The Make Visible command takes the object attributes affected by the selection rectangle and makes them visible in the schematic. This command applies only to attribute values (not pseudo-attributes like the part name) and only to visible objects. For example, it is not possible to display attributes associated with bus internal signals. |
| **Make Invisible** | The Make Invisible command takes the object attributes affected by the selection rectangle and makes them invisible in the schematic. |

### Disallowed Editing Changes

Any editing change that may cause a structural change to the connectivity of the schematic will be disallowed and aborted by the Browser. There are only a few special cases that will cause this to happen. For example:

- Renaming a visible signal to the same name as another visible signal on the same page would cause the two signals to become one signal.
- Making a signal visible which has the same name as an already visible signal with the same name would cause the two signals to become one signal.

## Saving and Printing Data in a Browser Window

The Browser does not have any commands for directly saving or printing data that appears in the spreadsheet display. However, the clipboard commands can be used to quickly transfer data to a word processor or text editor application for further processing.

To transfer the entire contents of the Browser spreadsheet to another application:

- Hold the Control while clicking and holding in the spreadsheet area of the Browser window. This will display the pop-up editing menu.
- Select the Select All command.
- Again holding the Control key while clicking and holding in the spreadsheet area of the Browser window, select the copy command.
- Switch to the desired destination application and select the Paste command in the Edit menu. This will place the Browser data in the current document in text format.

**TIP: If you will be making this kind of transfer frequently, we suggest creating a Scripter script to write the data to a file in the desired format. This will ensure consistent results and will not be dependent on the current settings in the Browser.**

### Browser Clipboard Data Format

The Browser exports data in a tab-delimited text format commonly used by spreadsheet and database programs. The following points summarize this format:

- Items appearing on one line in the spreadsheet display appear on one text line.
- Items on one line are separated by a tab character.
- Lines are terminated by a single carriage return character.
- If an item contains anything other than alphanumeric characters and spaces it is enclosed in double quotes. If it contains a double-quote character, that character is doubled, i.e. it is preceded by a second double-quote.

Note that attribute items can themselves contain carriage returns and other control characters. Such items will be enclosed in double-quotes in the clipboard text.

## Selecting Objects in the Schematic

The Select button will be enabled if there are objects selected in the spreadsheet. Pressing the button will cause the objects to be selected in the schematic but will not cause the schematic to be repositioned.

**NOTE: If the object type being viewed is Selected Device, Selected Signal, Selected Pin, or Selected Net then this operation will cause the spreadsheet to update to display the new selected objects.**

## Showing Objects in the Schematic

The Show button will only be enabled if there is a single object selected in the spreadsheet. Pressing the button will cause the schematic to be repositioned so that the object is visible. In addition, the object will also be selected and all other objects in the same circuit will be deselected. Note this will have the same side effect as above if the object type is Selected Device, Selected Signal, Selected Pin, or Selected Net.

The Show button may sometimes not be able to show the object. When this happens a dialog will be displayed. The reason for this is generally that the object was not in an open window.

## Opening Subcircuit Devices

The Open button will be enabled if there is a single object selected in the spreadsheet that has a subcircuit either directly or indirectly. Objects that may be opened are Devices and Pins (both for Pin objects and Pin connections for Net objects). Pressing the button will cause the subcircuit to be opened.

Sometimes it is not possible to open the subcircuit. When this happens a dialog will be displayed. The general reason for not being able to open the subcircuit is that a copy of the circuit is already open for another instance of the part.

## Sorting Displayed Objects

Pressing the Sort button causes the rows of the spread sheet to be sorted into increasing order.

If the object type is not Net or Selected Net then the following comparison mechanism is used when comparing two rows:

- The values of the first column are compared. If the values are not the same then this is sufficient to determine the sort order.
- If the values were the same then the next column is used to determine the sorting order. This is repeated as often as needed or until both rows are determined to have the same sort value.

If the object type is Net or Selected Net, then the rows are not sorted by the contents of the spreadsheet, but are instead sorted by the net's name that is in the row label.

## Adjusting the Spreadsheet

### Resizing Rows & Columns

Rows and columns may be resized by moving the cursor into the row or column heading area and positioning it over the horizontal (row) or vertical (column) dividing line. The cursor when correctly positioned it will change shape. Press the mouse button and drag the line to size the row or column to its desired size.

### Moving Rows & Columns

Rows and columns are reordered by moving the cursor into the row or column heading area and positioning it over the row or column heading to move. The cursor will change shape to indicate that it is correctly positioned. Press the mouse button and drag the label to its new position.

**NOTE: It can be useful to move columns so that the sorting mechanism will base the sort on the new ordering of the columns.**

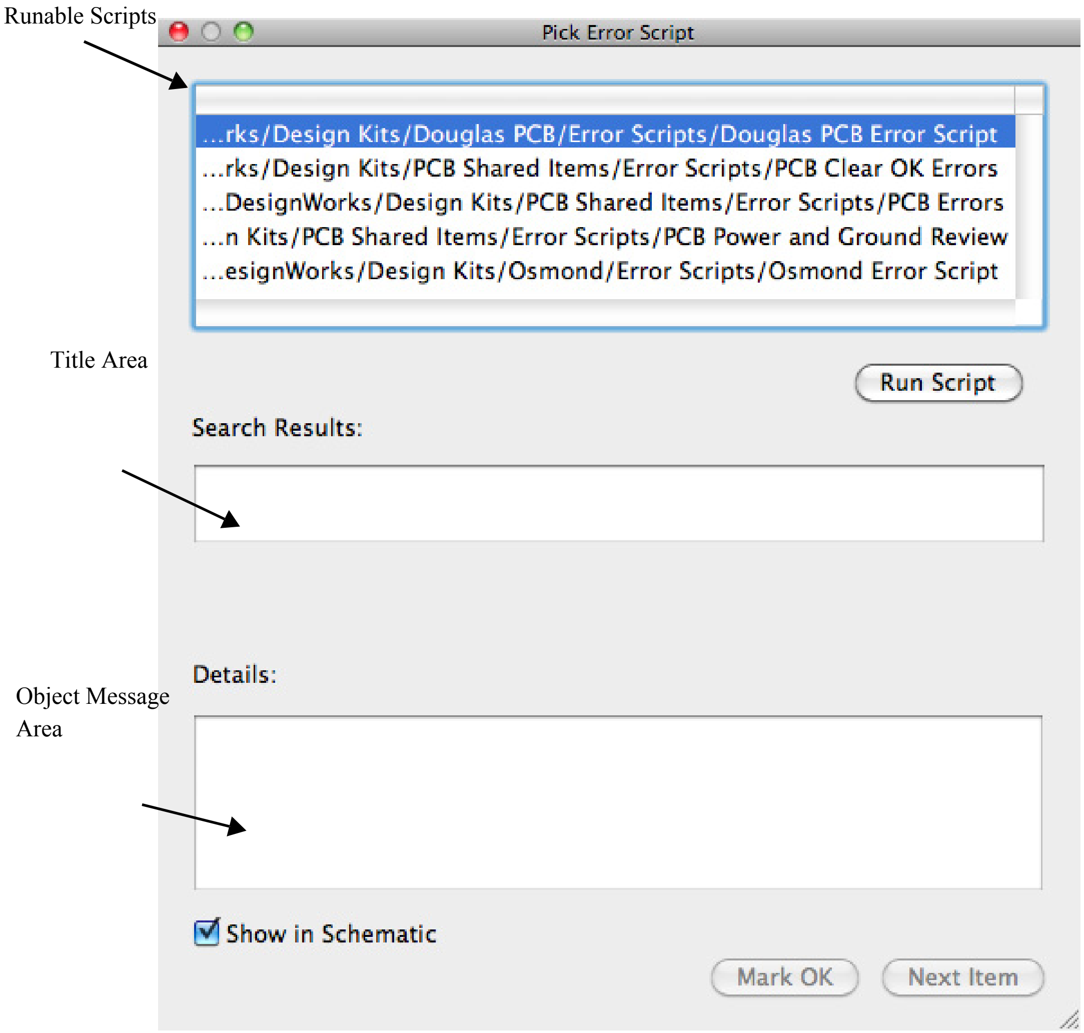
# Using the ErrorScript Tool

The ErrorScript tool runs a script file to locate circuit objects (devices, pins or signals) that meet some criteria, then allows you to locate and view these objects one at a time. If the script is set up for it, you can also use the "Mark as OK" feature to mark items that you wish to not be considered as errors in future searches. Although ErrorScript was designed primarily as an error location tool, the search script can actually use any available data about an object in generating its list. It is possible to use this facility to

## Running an Error Check Using ErrorScript

To use ErrorScript:

- Select the ErrorScript item in the tools menu. This will display the following script selection box:



...rks/Design Kits/Douglas PCB/Error Scripts/Douglas PCB Error Script
...rks/Design Kits/PCB Shared Items/Error Scripts/PCB Clear OK Errors
...DesignWorks/Design Kits/PCB Shared Items/Error Scripts/PCB Errors
...n Kits/PCB Shared Items/Error Scripts/PCB Power and Ground Review
...esignWorks/Design Kits/Osmond/Error Scripts/Osmond Error Script

☑ Show in Schematic

**NOTE: The contents of this box depends on the currently active design kits. In a standard installation, the items listed here will be the contents of the Error Scripts folders in all active design kits.**

- Select one of the scripts from the list and click the Run Script button. There will be a delay while the script executes, then results will appear in the bottom two text area's:

**NOTE: Many aspects of ErrorScript operation depend upon the logic of the selected script. Scripts can optionally display other data boxes, percentage complete, error messages, etc. The behaviour of an error script may vary from the typical case described here.**

The results box displays the script results for one object at a time. The title area is set by the script with overall results of the search. This usually includes the name of the results file, so you can open the file directly with a text editor in case you need a hardcopy listing of errors for later review. The object message area describes the error associated with the current object. This area will be updated each time you click the "Next Item" button to move to the next object.

### Locating Found Objects on the Schematic

If the "Show on Schematic" box is checked, the current object will be located on the schematic and selected each time the "Next Item" button is clicked. Some types of circuit objects may not be able to be shown:

Bus internal pins and signals have no visible manifestation on the schematic. If one of these is found by the script, its containing bus will be selected.

If the found object is inside a subcircuit block, the subcircuit will be opened and the object selected. This will not be possible in the specific case where the object is in a physical mode design and another instance of the containing subcircuit is already open for editing and has been modified.

### Marking Located Objects as OK

The "Mark as OK" button allows you to indicate that a found object is not actually an error and should not be found again by the same test. This allows you to keep "false alarms" out of the test results so that actual errors will be more easily located in future checks.

**NOTE: This feature may not be implemented in all error checking scripts. If the script does not generate the**

**information required, this button will be disabled.**

### Clearing the Mark as OK Settings

The Mark as OK settings stored with objects by previous error checks can be removed in either of these ways:

- Use ErrorScript to run a Clear OK Errors script provided with the design kit. This will not locate any objects, but it will clear the Mark as OK settings for all objects in the circuit.
- Manually remove any value from the OKErrors attribute field in the device, signal or pin in question. The numeric value stored by ErrorScript is taken as a set of bits that mask out specific errors. Setting the value to "0" or removing it completely indicates that all error checks should be performed on this object again.

## Customizing ErrorScript Search Scripts

ErrorScript has no ability to locate circuit objects and examine properties on its own. It reads a report file generated by the Scripter in a specific format which lists the objects of interest. It reads this file and then displays the results to the user in an interactive manner. Creating an error script thus requires some knowledge of the particular format read by ErrorScript and the scripting techniques required to generate such a file.

### ErrorScript Data File Format

The ErrorScript module reads a text file containing a list of objects to locate for the user. For each object, the following information can be supplied:

- The object's locator string. This is used to uniquely locate the object in the schematic, even in hierarchical designs. This is required.
- A text message to be displayed when this object is located. This will normally indicate to the user the type of error found and any other relevant information. This is technically optional, although the user will be in the dark about why this object is selected if no text is provided.
- An error bit number that can be set to mask out this error. This is used to implement the "Mark as OK" function on the ErrorScript control panel. If the user clicks this button, the given error bit will be set in the OKErrors attribute field. It is the script writer's responsibility to

check this bit when looking for errors and to skip over this object if it comes up.

In addition to the information provided per object, the file can also contain a title line that places a title in the ErrorScript results box. This will normally indicate the type of error check done, the number of errors found, and other status information.

**NOTE: Error Script developers should note that if you use the Mark as OK function in your scripts, you should provide a "Clear OK Errors" script to clear these settings in all marked objects. There is no user interface provided in ErrorScript for clearing these flags.**

### Data File Example

The following text illustrates a simple ErrorScript data file.

**NOTE: Although the format of the command keywords is similar to a Scripter command file, an ErrorScript data file <u>is not</u> a script file. The Scripter will normally be used to generate the data file, as described in "Generating an ErrorScript Data File Using the Scripter" on page 171.**

```
$TITLE Douglas PCB error check found 2 errors.
$OBJECT D23D
$MESSAGE Device U1*4 has an invalid name.
$ERRBIT 12
$OBJECT D25D
$MESSAGE Device U17 has one or more invalid pin numbers
$ERRBIT 14
```

This table describes the keywords used in the data file:

| | |
|---|---|
| **$TITLE** | This sets the title displayed at the top of the ErrorScript results box. It should normally be set once at the beginning of the file, although ErrorScript will update the title any time this item is encountered. The title text starts immediately following the $TITLE keyword and can continues until any other $ keyword is found. This means that the title text can be multiple lines, obviously limited by the space available in the output box. |

| | |
|---|---|
| **$OBJECT** | This indicates that start of the data for a new object. Following the keyword must be a valid object locator string. The locator will be used to find the object in the design when the user elects to show this object. The locator string is generated using the $DEVLOC/$SIGLOC/$PINLOC script commands. |
| **$MESSAGE** | This sets the message displayed in the message box in the ErrorScript results window. The format is the same as for $TITLE, that is, the message text starts after the $MESSAGE keyword and continues for any number of lines until the next $ command is found. |
| **$ERRBIT** | This sets the bit number used to mask this error using the "Mark as OK" function. If this is not supplied, the "Mark as OK" button will be disabled. This must be a decimal integer between 0 and 31. |

Note that all keywords must start at the beginning of a new line and must appear with all upper case letters, exactly as shown.

## Generating an ErrorScript Data File Using the Scripter

When the user selects a script from the list in the initial ErrorScript window, these actions take place:

- ErrorScript passes the name of the script to Scripter and asks it to execute it.
- Scripter executes the script which must generate a data file in the format described earlier in some known disk location and must pass the full name of the file, including directory path name, back to ErrorScript.
- ErrorScript opens the data file and locates the first selected object.

The execution of the Scripter in this mode is a bit different than operating it directly from the Tools menu. In this case, when the script starts executing, there is already a "report output file" open which is actually a memory buffer in ErrorScript. Any output generated by the script will be interpreted by ErrorScript at the file name. The script must contain an explicit $CREATEREPORT command to generate the disk data file, followed by a $CLOSEREPORT, followed by commands to generate the name of the file to pass back.

Here is a simple example to look for devices with no package code that illustrates these points:

```
$CREATEREPORT($TEMPPATH\package report)

$FIND $DEVICES $NOT(&Package)

$TITLE $DEVCOUNT devices found.  The error file is in $TEMPPATH\hello file.

$DEVICES\$OBJECT $DEVLOC$NEWLINE\$MESSAGE Device $DEVNAME has no Package
code.

$CLOSEREPORT

$TEMPPATH\package report
```

Note these points:

- The $ keywords required by ErrorScript in the output file should always be prefixed with an escape character "\" so that Scripter does not attempt to interpret them as commands.
- You must be careful to ensure that only a single line containing the file name is generated outside the $CREATEREPORT/ $CLOSEREPORT pair.
- Everything between $CREATEREPORT and $CLOSEREPORT is output to the data file. The format of this file must meet the specifications given earlier.
- It is best to specify a complete directory path name when creating the data file and describing its location to ErrorScript. This will ensure that the file does not end up in an unexpected location due to the setting of the system's "current directory". You can use the script keywords $DESIGNPATH, $SCRIPTPATH, $TEMPPATH and $PROGPATH to generate complete path names to known locations without specifying the absolute name of your disk. Note that $DESIGNPATH will be null if the design has not been saved.

As long as you keep these points in mind you can use any of the Scripter capabilities outlined in the DesignWorks Script Language Reference (separate manual on disk) to produce the data file.

# Attributes

# 9

This chapter describes the methods in DesignWorks for entering, editing, displaying and using attributes. Attributes are arbitrary blocks of text that can be associated with any device, signal or pin in a design, or with the design itself. Attributes have a wide variety of uses, including:

Displaying device name, package assignment, part type, value, etc. on the schematic.

Storing manufacturing information such as catalog numbers, manufacturer's name, price, part description, etc.

Storing data for use by external systems such as simulators, PCB layout, analysis tools, etc.

Attributes can be used in conjunction with the scripting and report generation features of DesignWorks to implement powerful interfacing, customizing and error checking features. You can define as many attribute fields as you need to store the text data required for your application.

In addition, many DesignWorks features use attributes as temporary or permanent storage locations for data relating to a device or signal. For example, each device in a design has the name and location of the library it was read from stored in an attribute field. This allows the library to be located easily for later updating.

## Attribute Organization

### Attribute Definition Table

Attributes are divided into named *fields*. The list of allowable field names is stored with each design in the *attribute definition table*. For each field, the table contains:

the field name (up to 16 characters)

what types of objects that field can be used in (devices, signals, pins or design)

data entry constraints (maximum length, carriage returns, etc.)

default settings, such as whether the field should be made visible on the diagram.

Once a field is defined in the table, its name is never typed again. When attribute data is entered for a selected object, the field name is picked from a list of allowable fields. This eliminates any chance of accidentally misspelling the field name in one object.

## Predefined Fields

When a new design is created, a default attribute definition table is created with the standard, predefined fields. Predefined fields cannot be deleted from the design since many of them have specific internal functions in DesignWorks.

***See Appendix C - Predefined Attribute Fields for a complete listing of predefined fields.***

## User-defined Fields

You can add more fields to a design at any time using the Define Attribute Fields command in the Options menu.

## Primary vs. Secondary Fields

DesignWorks has 40 or more predefined fields that are used for various program features and more can be added by the user. Most of these fields never need to be viewed or set directly by the user. To reduce clutter, attribute fields can be defined as being *primary* or *secondary*. When entering attribute data, you can choose whether you wish to see "primary fields only" or "all fields". The Attribute Probe tool ( ? ) in the tool palette shows only primary fields.

The primary/secondary setting is for convenience only and has no effect on the meaning or internal usage of the field. This setting can be changed at any time using the Define Attribute Fields command in the Options menu.

## Definition vs. Instance Fields

In a hierarchical design, a single device, pin or signal in a sub-circuit may actually represent several physical objects. This happens if the parent device containing the sub-circuit is used more than once in the design.

For this reason a distinction is made between fields that are associated with the definition of a sub-circuit and fields associated with the instance.

> Fields associated with the definition will be the same in all instances of the sub-circuit. Changing the value in one instance will affect all instances equally.
>
> Fields associated with the instance can take on a different value in each physical instance of the device, pin or signal. Changing the value in one instance has no effect on the others.

When defining a new attribute field for use in a hierarchical design, careful consideration should be given to whether to consider it a definition or instance field. If the value of the field has a significant impact on the logical functioning of the sub-circuit, then the field should probably be a definition field. Thus, if two sub-circuits require different values in this field, then separate definition circuits will have to be created. This makes it clearer to other users of the design that a distinction exists.

***See Chapter 11—Hierarchical Design on page 235 for more information.***

## Attribute Limitations

Attribute fields have the following specific limitations:

> Maximum length of field name: 15 characters
>
> Maximum length of field data item: 32,000 characters. Note: In principle, binary data could be pasted into an attribute field although this is not recommended since DesignWorks provides no mechanism for displaying or editing this data.
>
> Maximum number of user-defined fields: 900
>
> Maximum number of displayed positions of a single attribute item: 100

Like all other circuit data, the amount of attribute data that can be associated with a design is limited by available memory.

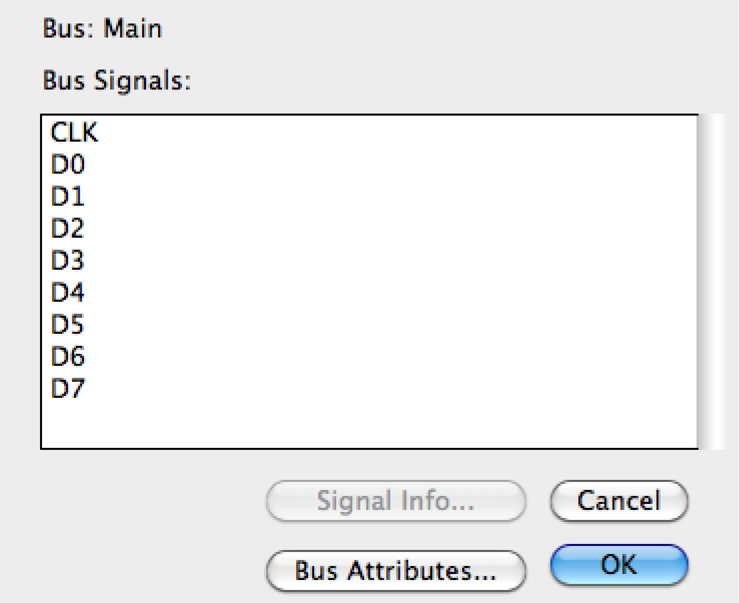
## Entering and Editing Attribute Data - Basic Procedure

To edit the attributes associated with a device, pin or signal, you can either:

> Select the object of interest on the schematic (you can only enter data for one object at a time using this method), then select the Get Info command in the Options menu. In the info box that is displayed, click on the Attributes button.
>
> Display the pop-up menu by holding the control key while clicking on the object. Select the Attributes command in the menu.

To edit design attributes, select the Set Design Attributes command in the Options menu.

Any of these methods will display the general attribute editing box which is used to enter or edit attribute data on a single object:



By default the field list at the left side of this box shows only the primary fields available for the selected kind of object. You can click on the All fields button to view all available fields. Custom fields can be added to this list using the procedure outlined in “Defining a New Attribute Field” on page 192.

## Entering Data Values

To edit the contents of a field, simply select the field name in the list. The current contents of the field will be displayed in the editable text box. Edit this value using the usual Macintosh text editing techniques. Press the OK button or select another field when done. When you select another field, data entered for the previous field is saved. If the data you typed exceeded the maximum length for the field, or if it contained invalid characters for the field, then you will be asked to correct the data.

You can view or edit as many fields as desired while in this box. No changes are made to the actual design data until you click the OK button. Clicking Cancel will abandon all changes made while in this box.

### Restoring the Default Value

Clicking the Use Default Value button sets the value for the selected field to the default value stored with the symbol. This is valid only for device and pin attributes for which a default value was specified. If this button is inactive (grayed out) then the value is already the default value, or no default value is present. Only devices and pins can have default values.

***See "Setting Default Values" on page 191 for more information on entering default attribute values while editing a symbol.***

### Using the Value List Pop-up Menu

If the selected field has an associated value list field, then a pop-up value menu will appear just below the "Use Default Value" button. An item selected out of this menu will cause the value text box to be updated accordingly. This does not prevent the text data from being edited directly. See more information on value list fields in "Using Value List Fields" on page 188.

### Displaying an Attribute on the Schematic

To display device, signal or pin attribute text on the schematic:

Edit the attribute value as desired.

Turn on the Visible switch.

When you click OK, the attribute text will now be displayed in a default position near the device, pin or signal. It can be dragged to the desired location using the pointer cursor.

***Design attributes can be displayed on the schematic using***

***text variables. See "Using Text Variables" on page 110.***

***For information on setting the default position for an attribute on a symbol, see "Using Default Position Fields" on page 190.***

## Entering Design Attributes

To view or set attributes associated with the design itself, select the Set Design Attributes command in the Options menu. This command displays the standard attribute data entry box allowing you to enter attribute data for the design itself. Design attributes can be used to create variables in text blocks on the schematic and to insert global data in netlist output.

**NOTE: The Visible option is not available for design attributes, since there is no single object on the schematic to associate them with. Design attributes can be made visible by using text variables, descrided in *"Using Text Variables" on page 110*.**

***See the entry Attribute Field References in the DesignWorks Script Language Reference (separate manual on disk) for more information on design attributes in netlists.***

## Entering Pin Attributes

To enter or edit attributes for a pin, you can use any of the following methods to display the attribute editing box:

Select the pin of interest on the schematic (make sure you are selecting the pin and not the signal, see "Selecting a Pin" on page 98), then select the Get Info command in the Options menu. In the pin info box that is displayed, click on the Attributes button.

Display the pop-up menu by holding the control key while clicking on the pin. Select the Attributes command in the menu.

Select the device that the pin is attached to, select the Get Info command to display the device info box, then click the Pin Info button in this box, then the Attributes button in the pin info box.

The attribute editing box for pins is identical to the one for devices, signals and the design, except that Next and Previous buttons appear at the bottom. These can be used to view and edit other pins on the same device without having to return to the schematic and select them individually.

# Controlling Attribute Display Characteristics

## Rotating Attribute Text

To rotate an attribute text item that is already displayed on the schematic, follow these steps:

Hold the control key while clicking and holding on the text item. This will display the attribute pop-up menu.

Select the Rotate Right or Rotate Left command.

## Hiding a Visible Attribute Value

If you wish to hide a device, pin or signal attribute value that is displayed on the schematic (that is, remove the displayed text from the schematic, but keep the value associated with the object), you can use any of these methods:

Hold the control key and click and hold on the text. In the attribute pop-up menu that appears, select the Hide command.

Hold the control key and click and hold on the text. In the attribute pop-up menu that appears, select the Edit command. In the text edit box that appears, turn off the Visible switch.

Display the general attribute editing box for the object, using the methods described earlier in this chapter. Select the field in question and turn off the Visible switch.

Click on the ⚡ tool in the tool palette, or select the Zap command in the Edit menu and then click on the attribute text you wish to hide.

Any of these methods removes the displayed text from the schematic without modifying the value that is stored with the object. The value can still be used for report generation, etc.

**NOTE: If the value you are hiding is a signal name, this operation may have side effects on the connectivity represented by the signal. See "Signal Connectivity Rules" on page 232 for more information.**

**Design attributes can only be displayed on the schematic using text variables. These are described in *"Using Text***

***Variables" on page 110***.

## Clearing a Visible Attribute Value

If you wish to remove the displayed attribute text associated with a device, pin or signal and remove its value completely, you can:

Hold the control key and click and hold on the text. In the attribute pop-up menu that appears, select the Clear command.

Hold the control key and click and hold on the text. In the attribute pop-up menu that appears, select the Edit command. In the text edit box that appears, remove the text value.

Display the general attribute editing box for the object, using the methods described earlier in this chapter. Select the field in question and remove the text value.

**NOTE: The first method above (the Clear command) actually performs a slightly different function to the other methods. It removes the visible text on the schematic and sets the attribute to its default value. For devices and pins, this means any value that was specified with the device symbol when it was created.**

## Displaying an Invisible Attribute Value

To display an attribute value that is already associated with a device, pin or signal, display the general attribute editing box using the methods described in "Entering and Editing Attribute Data - Basic Procedure" on page 175, then turn on the Visible switch. The value will be displayed in a default location near the object and can then be relocated as desired on the schematic.

Design attributes can only be displayed on the schematic using text variables. These are described in ***"Using Text Variables" on page 110***.

**NOTE: The Name attribute is a special case in that it can be displayed by simply clicking the text cursor on the object in question, in addition to the method described here.**

## Setting Attribute Text Style

Attribute text style is set globally for the entire design. There is no way to set text style for an individual attribute item.

**WARNING: Changing the attribute text style affects all visible attributes throughout the design. This may alter text alignment and position to accommodate a different text size.**

The attribute text style setting affects the following types of items:

Device and signal names.

Bus breakout and bus pin labels.

All other attributes displayed on the diagram.

but does not affect:

Border text (see "Setting Border Text Style" on page 360).

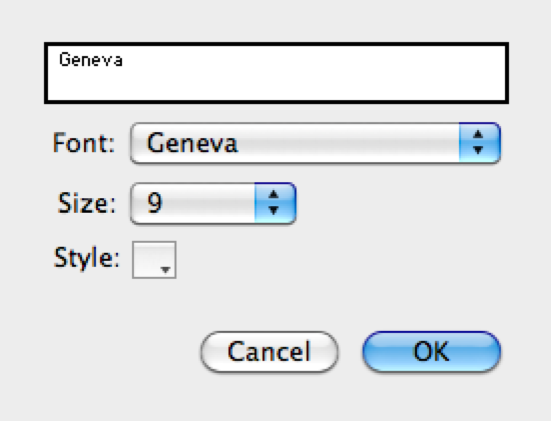
Pin numbers (see "Setting Pin Number Text Characteristics" on page 109).

The text changes are applied when the OK button in the Design Preferences box is clicked. For larger designs, there may be a substantial delay while new positions of all displayed attribute items are calculated.

To set the global text style:

Choose the Design Preferences command in the Drawing menu.

Click on the "Attr Text..." button. This will display the following box:



Geneva

Font: Geneva

Size: 9

This box allows you to select the text font, size and style used for all attributes displayed in the design.

Select the desired font, size and style, then click OK.

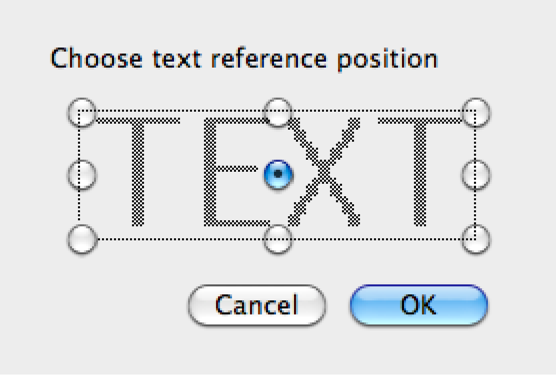
Click OK on the Design Preferences box.

Depending on the size of the design, there may be some delay at this point. The program must check all visible attribute items to see if there position and framing is affected by the text change.

## Setting Attribute Justification

Attribute justification determines how text is aligned relative to the device it is associated with. Generally, the program chooses a justification that is appropriate for the initial placement of the text and there will be no need to change it. However, if text is being modified frequently and does not reposition itself as desired, the justification can be modified individually for text items.

To set the justification, hold the control key and click and hold on the text item in question. In the attribute pop-up menu that appears, select the Justification command. This command allows you to change the vertical and horizontal justification used in the positioning the attribute text on the diagram. When this command is selected, the following box will be displayed:



The selected point on the text is considered to be the reference point for the given attribute block. This point will be kept fixed if any field value or text style changes cause the box to be resized.

**NOTE: The Justification command affects only the position of the text relative to the associated object. It <u>does not</u> affect the alignment of text in multi-line text items on the**

**schematic. DesignWorks does not provide any method of choosing center or right justification of multi-line text items.**

### Displaying an Attribute Value in Multiple Locations

You can display a single attribute value in multiple locations on a schematic. This is intended for use particularly with signals, which can be spread over large areas of a page or multiple pages. If a value is displayed more than once, editing any one of the displayed values will affect all appearances of it.

To create multiple displayed attribute items, hold the control key and click and hold on the text item in question. In the attribute pop-up menu that appears, select the Duplicate command.This command creates another visible occurrence of the same attribute field. This text can then be dragged or rotated to any desired position on the schematic.

### Showing the Field Name with an Attribute Value

The Show Field Name command allows you to display the field name with the value on the schematic. When this item is checked, the display will be in the form "fieldName=value". Selecting this command again will cause the display to revert to the normal value display. This command applies only to the selected field on the selected object.

**TIP: You can elect globally to show the field name for a certain field whenever it is used. See “Defining a New Attribute Field” on page 192 for more information.**

## Other Ways of Viewing and Editing Attributes

### Editing Attributes on the Schematic

Any attribute value associated with a device, pin or signal that is displayed on the schematic can be edited right in place using the text tool.

To use the text tool, click on the A item in the tool palette, or select the

Text item in the Edit menu. When you click the pencil cursor on any displayed attribute value, a flashing cursor will appear in the text to indicate that it is editable.

**NOTE: DesignWorks does not support in-place editing of attributes that appear rotated on the schematic. If you click in such a value, a simple value editing box will be displayed instead of the edit cursor on the schematic.**

An alternative method of editing a single displayed attribute field is to hold the control key and click and hold on the text. In the attribute pop-up menu that appears, select the Edit command. This command opens a text box allowing you to edit the contents of the selected field. All locations where this field is displayed on the schematic will be updated when the OK button is clicked.

## Using Value List Sub-menus

A device or pin pop-up menu may contain one or more sub-menus as its last items. These sub-menus allow you to select from a list of possible values for an attribute field. Selecting an item in one of these lists changes the value in the associated attribute field.

Most of the standard libraries provided with DesignWorks contain fields defining the part code, the package type and the gate unit (e.g. a, b, c, etc.). When one of these fields has more than one possible value, the list is displayed in one of these menus. For example, if the sub-menu "Unit" appears, this means there are multiple gate units available in this device type. Selecting one of the units in the Unit sub-menu causes the Unit attribute field to be changed for that device and the pin numbers to updated accordingly.

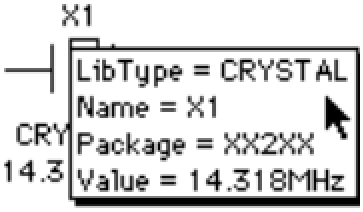
See “Using Value List Fields” on page 188 for information on creating your own value list fields.

## Probing Attributes on the Schematic

The contents of all fields associated with an object can be viewed with a single click using the Attribute Probe tool.

Select the Attribute Probe command in the Edit menu or click on the ? tool in the tool palette.

Click and hold on the device, signal or pin in question. The dot at the base of the question mark is the "hot spot" that determines which item is selected.



LibType = CRYSTAL
Name = X1
Package = XX2XX
Value = 14.318MHz

DesignWorks follows these rules in displaying fields in the attribute probe pop-up window:

Fields are shown sorted by field name.

Only fields with non-empty values are shown and only the first 32 characters of the field value will appear.

By default, only Primary fields are shown, but this depends on the last setting of the “Primary Fields/All Fields” buttons in the general attribute editing box. To change this setting, using the Get Info or Attributes command on any object using the general attribute editing box, then select the Primary Fields or All Fields setting as desired. This will affect the attribute probe next time it is used.

## Using the Name and InstName Fields

In DesignWorks, the single most important attribute field is the "Name" field. It can be associated with devices or signals and is used to uniquely identify an object within its circuit. In PCB designs, the “name” is also the “package assignment” in that it denotes the physical device that a symbol on the schematic is assigned to.

Names may contain any letters, numbers, "_" (underscore) or "." (period) characters. They are restricted in length to at most 15 characters. The name associated with an object can be placed anywhere on the diagram and will be removed if the object is removed.

**NOTE: DesignWorks does not actually prevent you from using any combination of characters in a name, including blanks or**

**special codes. However, many of the error checking and netlisting scripts will flag unusual characters as an error, depending on the characteristics of the external layout system or simulator being used.**

The Name field can be entered and edited just like any other attribute, but because it is so widely used, there are some extra features to facilitate working with it. These are described in detail in the following sections.

**NOTE: If you plan to create hierarchical designs, you should undertand the distinction between the Name and InstName fields. This is described in detail in the next section. For flat designs, the InstName field is not required and all device naming can be done with the Name field.**

## Choosing Whether to Use Name or InstName

The distinction between the Name and InstName fields becomes important if you plan to use the physical hierarchy mode in DesignWorks to stored physical package assignments or other data.

> The Name is a *definition* field, meaning that it will have the same value inside all instances of the same type of sub-circuit. For this reason, Name cannot be used to store physical data that may be different in separate instances of a sub-circuit.
>
> The InstName field is used in hierarchical designs for PCB package assignments and other physical location data. InstName is an *instance* field, meaning that it can have a different value for every physical device in a hierarchical design. See the section "Names in Hierarchical Designs" on page 188. The InstName field is not normally used in Flat mode designs.

***You can find more information on hierarchical design modes in Chapter 11—Hierarchical Design on page 235.***

### Invisible Names

Like other attribute fields, the Name field can be associated with a device or signal without displaying it on the diagram. The visibility of the name can be set using the Get Info or Attributes command associated with the device or signal in question.

**NOTE: When signals are connected by name, only visible names affect connectivity. Two signals with the same invisible name will not be connected. See "Signal Connectivity Rules" on page 232 for more details.**

## Device Names

Typical device names might be "U23", "C4", "XTAL1", etc. The Name field for devices is equivalent to the "reference designator" used in other systems.

The Name is distinct from the "type name" that is used to distinguish the type definition that is read from a device library. The "type name" is not an attribute, but a fixed name associated with a device symbol when it is saved in a library. Typical type names are "74LS138", "MC68000L8", "SPDT Switch", etc.

As well as appearing as a text notation on the diagram, the device Name field is used by the following DesignWorks functions:

if the design is in flat hierarchy mode, then the Packager tool places the assigned package name in the Name field. In other modes, the "InstName" field is used.

the device name is used in report output, such as netlists and bill of materials reports.

the device can be located by name using the Find tool.

the device name associated with a Page or Port Connector is used to make logical connections.

## Signal Names

Typical signal names are "CLOCK", "ADDR12", etc. Signal name is referenced by the following DesignWorks functions:

the signal name is used in report output, such as netlists.

the signal can be located by name using the Find tool.

signals can be logically interconnected by name.

***See more information on signal names and logical connections in Chapter 4 - Schematic Editing.***

### Names in Hierarchical Designs

In a hierarchical design, a single device in a sub-circuit may actually represent several physical devices. This happens if the parent device containing the sub-circuit is used more than once in the design.

***See "Definition vs. Instance" on page 237 for an example of this situation.***

For this reason another attribute field is needed that will be unique for each and every physical instance of a device in the design. The "InstName" field serves this purpose.

If the design is in "physical" hierarchy mode, the InstName field is used by the Packager to store the package assignment. If the Packager is not being used, the InstName field can be used to store notations about the physical placement of the device.

**NOTE: In a hierarchical design, the Name field can be thought of as a "logical" name and the InstName field as a "physical" name. The "logical" name should be assigned by the designer based on the device's function in the circuit, not on its physical location.**

The visibility of the Name and InstName fields on the diagram can be set separately. If may be desirable to only show the Name field on the diagram during the early stages of design development. The physical package assignments can be shown later when needed for manufacturing.

## Using Value List Fields

A value list field allows you to specify a list of possible values that can be used in a selected field. This list of values if then displayed to the user at various locations, allowing easy selection of one of the available items. For example, this mechanism is used to create the "Unit" and "Part" fields used in all standard DesignWorks libraries. These standard fields can be used as an example.

**NOTE: DesignWorks does not restrict the user from entering values different from the ones in the list using the usual**

**attribute editing techniques. If you want to restrict the values available in a given field, you may wish to create an error checking script that verifies field values. See "Using the ErrorScript Tool" on page 166 for more information.**

A value list field has the same name as the associated field but with ".List" appended. When a field named "xxx.List" is seen by the program, it takes these steps:

> When the attribute edit box is displayed for a given object and the field "xxx" is selected, a pop-up menu is displayed containing all the possible values for the field. The user may then select the appropriate value from the menu.
>
> When a pop-up menu is displayed by the user using the control key, a sub-menu with the field name "xxx" as its title will appear. The list of possible field values will be displayed in the sub-menu. The user can select one of these items to directly change the value of field "xxx".

### Value List Data Format

The value list field must contain a sequence of textual values following these rules:

> Items must be separated by commas. A comma will always be taken as a separator and therefore cannot be part of the value.
>
> The value string can contain any characters other than commas, but leading and trailing blanks will be removed.
>
> The length of a value string is limited to 16 characters.
>
> The maximum number of values is limited to 256.

Normally, a default value for the "xxx.List" field should be defined with the symbol in the library using the DevEditor tool. In this way, the value list will always be available whenever this device type is selected from the library. If the field is defined in only one device on the diagram, the value list will only be available for that device.

## Creating a Value List Field

The creation of value list fields will be illustrated by way of an example.

Suppose you wish to create a symbol for a device which will be simulated using an external simulator. This symbol can be associated with one of three models which have the same pinouts, but different timing parameters. The model name is to be included in the netlist. To create a list of the available model values:

> Using the Define Attribute Fields command, create two new fields, called "Model" and "Model.List". For both fields, check the "Devices" box indicating that these will be associated with devices.
>
> Using the DevEditor tool, create the desired device symbol, or open an existing one for editing. Using the DevEditor's Part Attributes command, create a field called "Model.List" containing the desired list of values. For example:
>
> `MODEL1, MODEL2, MODEL3`
>
> Save the edited symbol to a library. Select the part from the library in the usual way and place it on a circuit diagram.
>
> Hold the control key and click on the device. You will now see at the bottom of the pop-up menu a new item called "Model". Moving down to this item will display a new sub-menu containing the values entered in the "Model.List" field.

## Using Default Position Fields

For device or pin attribute fields that are to be displayed on the schematic, an associated field called a "default position field" can be created. For example, this mechanism is used to provide default positions for the "Name" and "Part" fields used in all standard DesignWorks libraries. These standard fields can be used as an example.

A default position field has the same name as the associated field but with ".Pt" (for Point) appended. When a field named "xxx.Pt" is seen by the program, it uses the values in this field to generate a default position for the "xxx" field.

### Default Position Data Format

The default position field must contain a sequence of textual values following these rules:

> Two items, an X and a Y position, must be specified, separated by a comma. Leading an trailing non-numeric characters are ignored.

The X and Y values are each decimal numbers representing an offset from the top-left corner of the device measured in 1/1000". Negative values are allowed (negative is up and left).

The X coordinate can be optionally followed by a letter indicating horizontal justification, as follows: "L" for left, "M" for middle, "R" for right. I.e. the given number specifies the offset to the left, middle or right position of the displayed text. If no letter appears, middle is assumed, i.e. the text will be centered at the given position.

The Y coordinate can be optionally followed by a letter indicating vertical justification, as follows: "T" for top, "M" for middle, "B" for bottom. I.e. the given number specifies the offset to the top, middle or bottom position of the displayed text. If no letter appears, middle is assumed, i.e. the text will be centered at the given position.

Some examples of value points are:

| | |
|---|---|
| **0R,0T** | The value will extend right and down from the top-left corner of the device. |
| **150,-75** | The value will be centered at a position above and to the right of the top-left corner. |
| **(400L,100)** | The value will extend left from a point to the right of the top-left corner of the device. (Leading and trailing characters are ignored.) |

Normally, a default value for the "xxx.Pt" field should be defined with the symbol in the library using the DevEditor tool. In this way, the value will be available for setting the default position of the associated field when the device is placed on the schematic.

## Setting Default Values

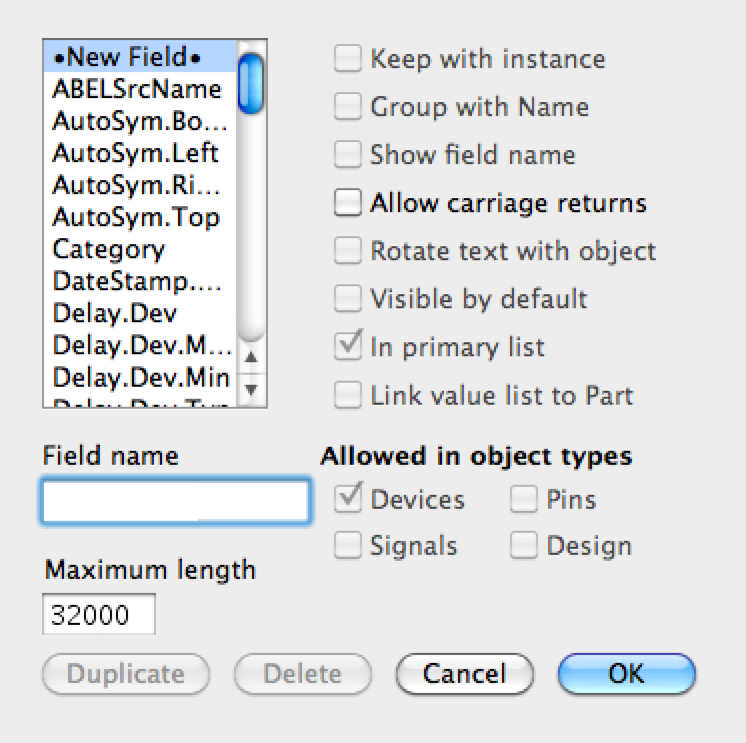
A device symbol can have associated with it predefined default values for any number of fields. Values can be specified for the device itself, and independently for each pin on the device.

When the standard attribute data box is displayed for a device, you will see a button labeled "Use Default Value". If this button is grayed out (disabled), then there is no default value or the value shown is already the default.

***See "Setting Part and Pin Attributes" on page 295 for more information on creating default attribute values.***

## Defining a New Attribute Field

All changes in field definitions are done using the Define Attribute Fields command in the Options menu. Selecting this command displays the following dialog box:



☐ Keep with instance
☐ Group with Name
☐ Show field name
☐ Allow carriage returns
☐ Rotate text with object
☐ Visible by default
☑ In primary list
☐ Link value list to Part

Field name

Maximum length
32000

**Allowed in object types**
☑ Devices ☐ Pins
☐ Signals ☐ Design

To create a new field:

Select the •New Field• item in the list, if it is not already.

Type the name of the field in the Field Name box.

Select one or more of the boxes under "Allowed in object types".

Optionally enter a maximum length.

Optionally enable any of the other field options. NOTE: These options affect only new data as it is entered, not values already in the design.

Click OK. The new field will now appear in the attribute data entry box for the selected object types.

## Setting Attribute Field Options

When a new field is created using the Define Attribute Fields command a number of options are available which determine how that field will behave in your design. These options are described in the following sections.

**NOTE: Changing any of the following options in a field that has already been used in the design has <u>no effect</u> on items already displayed on the schematic.**

**TIP: To make a temporary change (e.g. to display a field that is not normally displayed) you can use the Duplicate button in the Define Attribute Fields box to create a new field containing the same data but with new display options. This field can be deleted again when no longer needed.**

Note the following characteristics of these settings:

Changing the values of any of the option switches for an existing field do not affect any values already in the design. Only new data will be affected. See “Using Duplicate, Merge & Delete for Global Editing” on page 196 for tips on using the Duplicate function to update the way existing data is displayed.

The name and option settings for most predefined fields are fixed and will be disabled in this box.

Regardless of changes made in this box, no design data is updated until you click the OK button.

### Field Name

The Field Name box allows you to enter the name of a new field or rename an existing user-defined one. If you rename a field so that it has the same name as another existing field, this becomes a Merge operation described in “Merging Two Existing Attribute Fields” on page 198. You will be prompted to confirm a merge before proceeding.

### Keep with Instance

This option applies only in Physical Hierarchy mode. When this switch is checked, the selected attribute field may take on a different value for each instance of a device, pin or signal.

***See "Definition vs. Instance" on page 237 for more information on instance data.***

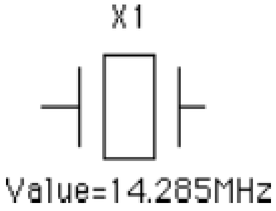
### Group with Name

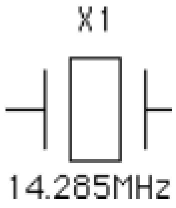
When this switch is checked, any value entered for this field will be displayed beneath the Name field by default. When the Name is moved, this field will also be moved.

This option can be used for fields that are typically shown with the name on a schematic. E.g. the Unit field (gate unit) has this setting enabled by default.

### Show Field Name

When this switch is checked, the field name will be shown with the value on the schematic.





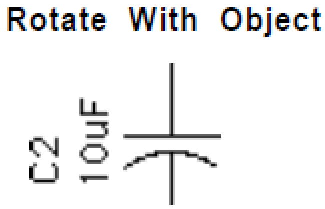
The Show Field Name option can also be controlled for each individual item on the schematic using the Show Field Name command in the attribute pop-up menu.

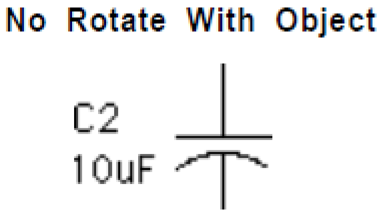
### Allow Carriage Returns

If this switch is checked, then the Return key may be used to enter Carriage Return characters into this field.

### Rotate Text With Object

If this box is checked, displayed values will be rotated if the associated object is rotated on the schematic.  Otherwise, when the object is rotated, the text will move to an appropriate new position but be drawn in the normal orientation.





### Visible by Default

If this box is checked, whenever a value is entered for this field it will be displayed on the schematic in a default position.  If the field is found in the default attributes for a device, the field will be displayed automatically on the diagram when the device is placed.

**NOTE: Changing this setting has no effect on values already in the circuit.  If you want to globally show all the values of a field on the schematic, see “Temporarily Displaying Attributes” on page 200.**

### In Primary List

If this box is checked, the field will be considered a "primary" field and will be displayed in all attribute entry boxes when "Primary Only" is selected.  It will also appear in the window that is displayed by the Attribute Probe ( ? ) cursor.

### Allowed in Object Types

This group of check boxes allows you to select which types of schematic objects may have this field associated with them.  You can check any one or more boxes, although in general we suggest using different names for fields used with different object types.

**NOTE: If you plan to place pin attributes in a design for later extraction in a report or netlist, you may wish to refer to the**

**entry "Precedence of Field References in Pin Listings in the DesignWorks Script Language Reference (separate manual on disk) for information on how pin attribute references are located.**

At least one of the "Allowed in Object Types:" switches must be selected for a new field.

### Maximum Length

You can specify a maximum length (in characters) for the field data. This value does not determine how much storage is allocated internally for the data, it only affects checks that are done when the user enters data. All attribute data stored with the design occupies only the space required for its current value.

### Duplicate

This button allows you to create a new field with the same settings and (optionally) data contents as an existing one. You will be prompted to confirm the data duplication option before proceeding. More information on this operation is given in the following section.

### Delete

This button allows you to permanently remove a field definition and all associated data from the design.

**WARNING: THIS CANNOT BE UNDONE!!!**

More information on the delete operation is given in the following section.

## Using Duplicate, Merge & Delete for Global Editing

The Duplicate, Merge and Delete Attribute functions do more than just add or delete a name in the attribute list. They will actually scan the entire design and update the data stored with each object. This makes them very powerful tools for updating your design. First we will review how the functions are invoked, then mention some possible uses for them.

These three functions are all invoked from the Define Attribute Fields command in the Options menu. No design data is modified until the OK button is clicked in the Define Attribute Fields box.

**NOTE: When the design is updated, all Duplicates are done first, then all Merges and Deletes. This means that a field can be duplicated, then the original source deleted, all in one invocation of the Define Attributes box.**

**IMPORTANT: There is no guarantee of the order of execution within the Duplicates or within the Deletes and Merges. Therefore, chains of operations (e.g. making a copy of a copy) are not guaranteed to work as expected. If there is any doubt, perform one operation, then exit the Define Attributes box using the OK button, then perform the next operation.**

**WARNING: Each of these operations can result in data being updated throughout a design. and they cannot be undone! For large designs there may be a considerable delay after the OK button is clicked in the Define Attribute Fields box.**

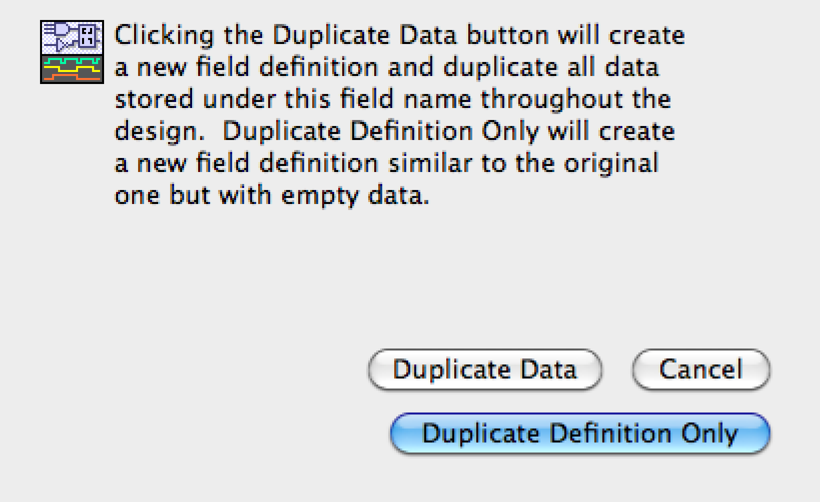
## Globally Duplicating Attribute Data

The Duplicate operation adds a new field name to the list with characteristics the same as an existing one. It will also optionally duplicate the data stored under that name throughout the design.

**NOTE: The pre-defined fields in DesignWorks cannot be renamed or removed. However, you can duplicate them, and the copy will be marked as "user-defined", meaning that it can be deleted again or modified in any way.**

A field is duplicated by selecting its name in the field list and then clicking on the Duplicate button in the Define Attributes dialog:

At this point a box will be displayed allowing you to choose one of the following options:



Clicking the Duplicate Data button will create a new field definition and duplicate all data stored under this field name throughout the design. Duplicate Definition Only will create a new field definition similar to the original one but with empty data.

| | |
|---|---|
| **Duplicate Definition Only** | A new field will be defined with characteristics just like the one being duplicated, but it will have an empty value in each of the objects in the design. |
| **Duplicate Data** | A new field will be defined and the associated value in each object in the design will also be duplicated and associated with the new field name. Changing the value of the new field in a given object will have no effect on the original field. |

One important feature of this Duplicate operation is that any attribute display options, such as "Visible by Default" or "Show with Field Name", that are turned on before leaving the Define Attributes box will be applied to the duplicated data. This means that a field that was formerly hidden can be duplicated and displayed throughout the entire design with just this one operation.

## Merging Two Existing Attribute Fields

The Merge operation takes two existing attribute field definitions and merges them into one. If a given object (device, signal, pin or design) has values in both of the original fields, the one being renamed takes precedence and the other is lost.

**NOTE: Pre-defined fields cannot be renamed, so they cannot be merged into other fields. You can however merge a user-defined field into a pre-defined one.**

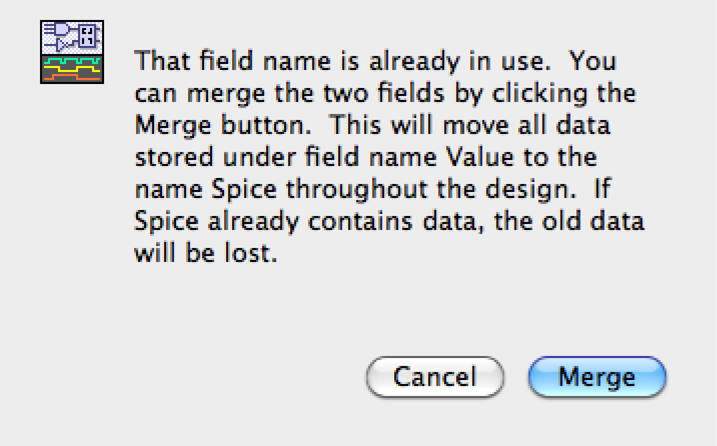
There is no explicit button for the Merge operation. Merge is invoked whenever a field is renamed so that its name matches another field's. The procedure is as follows:

Click on the field to be renamed in the field list. The data in this field will take precedence when the design is updated.

Type the new name (i.e. the name of the field you wish to merge to) into the Field Name text edit box.

Type the Tab key or click on any of the option switches to indicate that you have finished typing.

A box will be displayed asking you to confirm the merge. Click the Merge button:



That field name is already in use. You can merge the two fields by clicking the Merge button. This will move all data stored under field name Value to the name Spice throughout the design. If Spice already contains data, the old data will be lost.

The actual merging of design data does not take place until you click on the OK button in the Define Attributes box.

**NOTE: You can only merge fields that have similar "Allowed in Object Types" settings. E.g. you cannot merge a signal field into a device field.**

## Delete

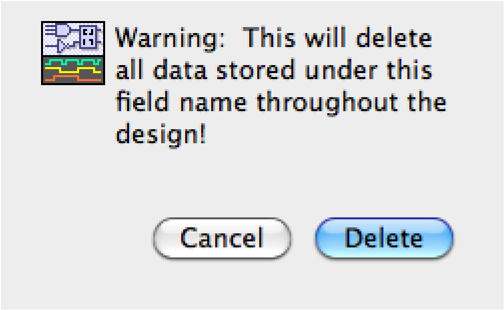
The Delete operation removes the selected name from the list of available fields and removes <u>all data values</u> associated with the field <u>throughout the</u>

design. Pre-defined fields cannot be deleted.

**WARNING: This operation cannot be undone and should be used with care.**

To delete a field, simply select its name in the field list and click on the Delete button:

The following box will be displayed to confirm the Delete operation:



Warning: This will delete all data stored under this field name throughout the design!

In any case, no design data is updated until the OK button is pressed on the Define Attributes box.

## Temporarily Displaying Attributes

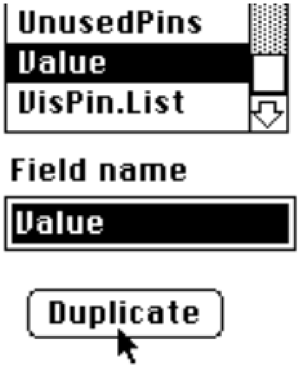
The Duplicate function is a convenient method of displaying data temporarily on the schematic. For example, you may wish to display some simulation parameters on the schematic while tracking a particular design problem, even though you don't want them there for the final printout.

To do this:

Select the Define Attribute Fields command in the Options menu.

Choose the field you wish to display.

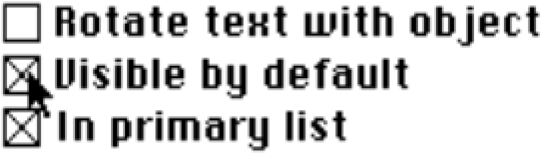
Click on the Duplicate button.



Field name: Value

Click on the Duplicate Data option in the confirmation box.

Enable the "Visible by Default" and (if desired) the "Show Field Name" and "In Primary List" options.



☐ Rotate text with object
☒ Visible by default
☒ In primary list

Click the OK button in the Define Attributes box.

The duplicate field will now be displayed on each object in the design. When the duplicate field is no longer needed, it can be removed with the Delete function described above.

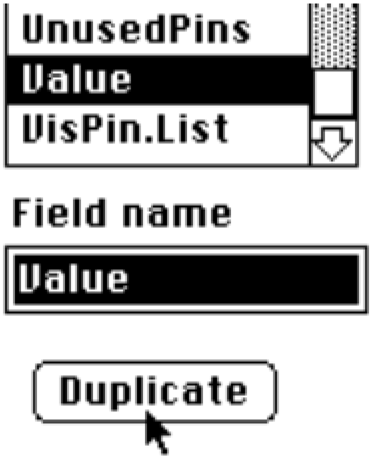
## Permanently Showing Data Throughout a Design

The "Visible By Default" option in the Define Attributes box normally has no effect on values that already exist in the design. Enabling the option for a field that is already in use affects only future entries. However, the Duplicate and Merge operations can be used to update the display of existing data throughout the design. This is done as follows:

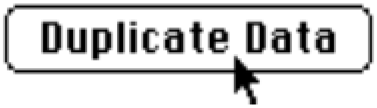
Select the Define Attribute Fields command in the Options menu.

Select the field to be displayed in the field list.

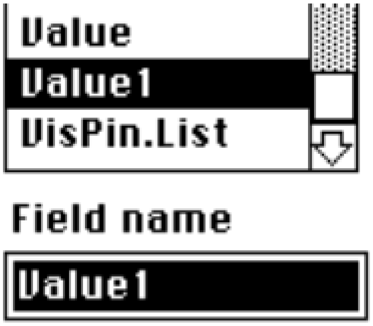
Click on the Duplicate button:



In the confirmation box, click on the Duplicate Data button:

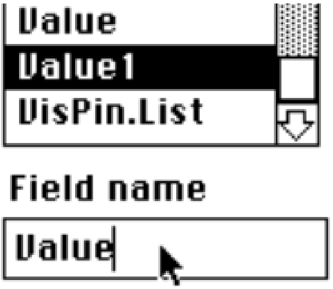


You will now have a copy of the original field:



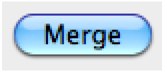
Field name: Value1

Make sure the newly-created field is selected in the list, then rename it back to the name of the original field. E.g. If the original field was "Value", the duplicate will be "Value1" and we now select "Value1" and rename it back to "Value". Press the Tab key when you are finished typing.

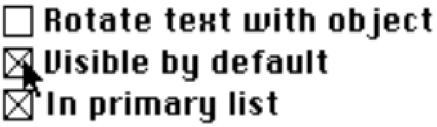


Field name: Value

The Merge confirmation box will appear. Click the Merge button.



Click on the "Visible By Default" option and any other display options desired.



☐ Rotate text with object
☒ Visible by default
☒ In primary list

Click the OK button.

This operation in effect re-enters all the data into the selected field with the new display options.

### Merging Dissimilar Designs

The Merge feature is a useful tool when bringing together designs created with different attribute definitions, or when importing designs from other systems or older versions of DesignWorks.

We strongly recommend reviewing attribute field usage under these circumstances and using the Merge feature to convert all non-standard fields to pre-defined ones where possible. This will make the design more compatible with the standard report formats provided with DesignWorks and will reduce confusion in future design revisions.

## Importing Attribute Definitions

Whenever circuit data is brought into a design, either from a design file, a device library or the clipboard, it is possible that the incoming data contains attribute fields that are not defined in the destination design. This section discusses how DesignWorks deals with this situation.

In general, DesignWorks attempts to ensure that an entry exists in the attribute definition table for every field used in the design. When new data is imported, field usage in the imported data is compared to the existing table. Normally, no discrepancies are found and this process is invisible. If any mismatch is found, you will be warned and given a chance to add the new definitions to the table automatically.

### Pasting from the Clipboard or Placing a Library Device

In these two cases, DesignWorks does the following:

All fields used in the incoming device or circuit scrap are looked up in the design's attribute table.

Any field that doesn't exist in the design is put on a list to be added.

Any field that does exist in the design but is incompatible (i.e. different "object type" or "keep with instance" settings) is renamed by adding an underscore "_" to the name.

A list of the new fields that will be added is displayed in a warning box. If any of these fields are unexpected, or can be converted to standard fields, the names should be noted at this point. Changes in field usage can be made using the Merge and Delete operations mentioned elsewhere in this chapter.

## Converting Files from Older Versions

DesignWorks versions prior to 3.0 did not have a standard list of pre-defined fields. For this reason, files converted from older versions should be reviewed to ensure that field names are consistent (i.e. spelling and capitalization variations are eliminated) and that standard names are used wherever possible.

The Converter tool makes a number of field translations automatically for commonly-used fields. See the Converter documentation for more information.

## Changes in Standard Fields

In rare cases, variations in the list of pre-defined fields will occur between versions of DesignWorks. This will normally only come up if you have used a pre-release or custom modified version of the package. DesignWorks checks for conflicts as it reads a design file into memory. If a field defined in the file is marked as pre-defined, but does not exist in the internal table, or vice versa, you will be warned when the file reading process is completed. The attribute table is automatically updated to resolve the conflict and resaving the file will update it to the new format.

# Making Signal Connections

# 10

This chapter provides more information on the advanced features available for making signal connections in DesignWorks. "Creating and Editing Signals" on page 93 covered the basic procedures for connecting pins together by signal line and joining signals by name. This chapter will cover these additional methods of describing signal connections:

Cross-page Connections and Automatic Page References

Busses

Bus Pins on Hierarchical Circuit Blocks

Using Power and Ground Connector Symbols

Specifying Power and Ground Connections in Attributes

Signal Auto-naming

Signal Token Numbers

Signal Connectivity Rules

## Who Makes the Rules Anyway?

DesignWorks has a set of rules that it applies to determine if a logical connection is made between two signals with the same name in different parts of a design. These rules are outlined in "Signal Connectivity Rules" on page 232. However, these may not be the only rules that apply, depending on where you are going with the schematic data. If you are exporting data via a netlist to another package for layout, simulation or analysis, that software will apply its own rules about connections between nets. Some systems (for example, SPICE-based simulators) may assume that two signals with the same name are connected together, whereas others base connectivity strictly off the structure of the netlist and make no assumptions about names.

The safest course is to ensure that you never use the same signal name twice and to check this using the duplicate signal name checking scripts provided with the package. In any case, it is wise to know the assumptions being made by all the packages that will be receiving your design data.

# Using Busses

The bussing facility in DesignWorks allows any combination of named signals to be represented by a single line and any subset of these to be brought out through a "breakout" at any point along the bus line.

## Properties of Busses

A bus is treated by DesignWorks as a signal with special properties. Thus, bus lines can be drawn and modified on the screen using all the same editing features available for signals. Note these properties of busses:

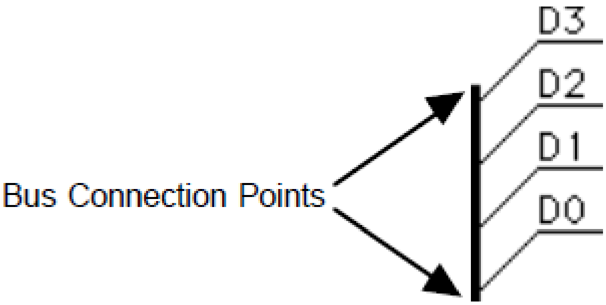
> Only bus pins on devices can be connected directly to a bus. All other connections must be made by using a breakout to access the desired internal signals. A breakout is created using the New Breakout command in the Options menu.
>
> You do not need to specify in advance what signals will be contained in a given bus. Any signals that are present in a breakout or bus pin attached to a bus will become part of that bus and can be brought out through another breakout anywhere along the bus.
>
> Any two busses can be joined together, regardless of their internal signals. When two different busses are merged, any signal in either bus becomes available anywhere along the combined bus.
>
> If you select a bus line, then do a Get Info command from the Options menu the displayed info box will show a list of the signals currently contained in the bus.
>
> A given signal can be present only in one bus. If you attempt to connect together two signals in different busses, a warning box will be displayed and the connection will be canceled.
>
> A bus can be created by drawing the bus lines first, then creating the breakouts to attach, or by creating a breakout and extending the bus line starting at the bus pin. Bus lines are drawn or extended using exactly the same techniques as for signals, except that the Draw Bus command or cursor is used instead of Draw Signal.

## Creating a Bus

A bus can be created by any one of these methods:

Select the Draw Bus tool (✚) in the tool palette. Draw any desired contiguous set of lines on the diagram using the usual signal drawing techniques. This bus will have no internal signals initially. Signals will be added implicitly when it is connected to any breakout or bus pin.

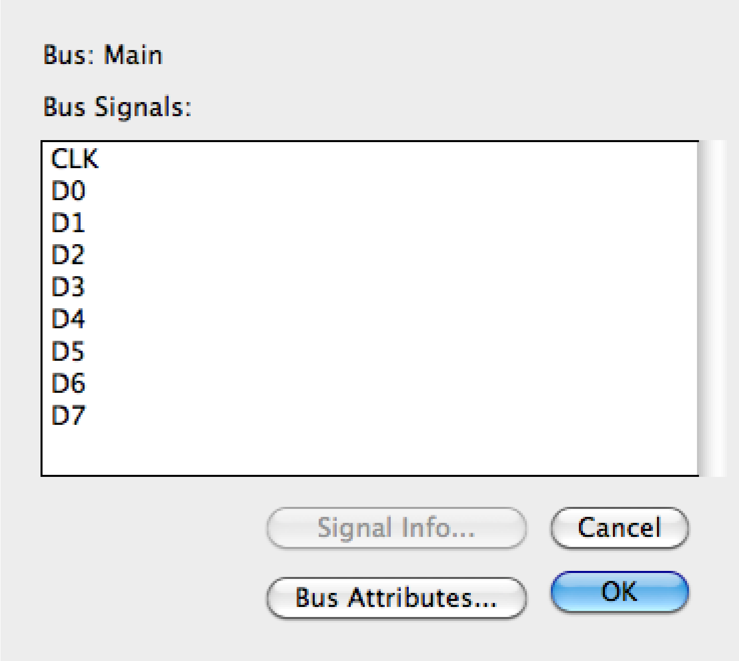
Create a breakout symbol using the New Breakout command discussed on page “Using Bus Breakouts” on page 211. The bus pin (backbone) of the breakout can now be extended using the normal pointer or the Draw Bus cursor. The bus will contain all signals specified in the breakout.



Extend a line out from an existing bus pin on a device using the normal pointer or the Draw Bus cursor. The bus will contain all signals specified in the bus pin on the device. Connections between bus internal pins and bus internal signals can be changed using the Bus Pin Info command on the bus pin's pop-up menu. See “Using Bus Pins” on page 214 for more information on bus pins.

## Getting Bus Information

A list of internal signals can be seen by selecting a bus and using the Get Info command. This action will display the following box:


Bus: Main

Bus Signals:

CLK
D0
D1
D2
D3
D4
D5
D6
D7

Here is a summary of the options presented in the bus information box:

| | |
|---|---|
| **Bus Signals** | This is a list of the signals contained in the bus. This list is determined by the breakouts and bus pins attached to the bus. You cannot directly change this list. Doubling-clicking on any item in this list is equivalent to clicking once and then clicking the Signal Info button. |
| **Bus Attributes** | This button displays the general attribute data entry box for the selected bus. NOTE: Except for the Name field, bus attributes are generally not included in any netlist output. More information on the functions available in this box are given in “Entering and Editing Attribute Data - Basic Procedure” on page 175. |
| **Signal Info** | This button displays the signal info box for the signal selected in the list. This allows access to signals that may not be visible on the schematic. More information on this box is provided in “Getting and Setting Signal Information” on page 98. |

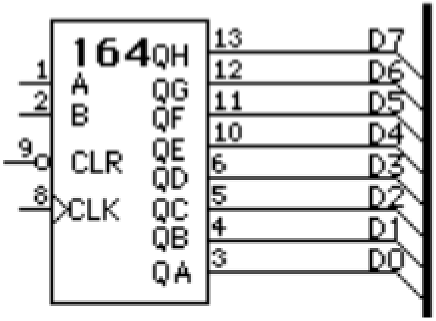
## Adding Signals to a Bus

There is no explicit command to add signals to a bus. Signals are added to a bus each time a breakout or device bus pin is connected to the bus.

Any signals in the breakout or bus pin are implicitly added to the bus if they don't exist already.

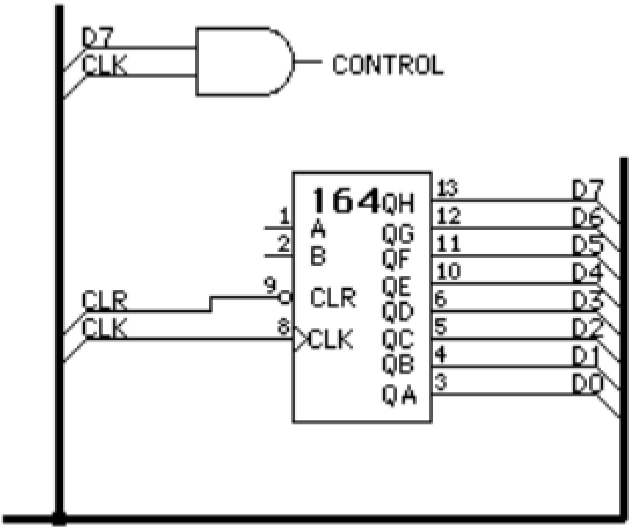
## Using Bus Breakouts

Signals are attached to a bus via a special type of pseudo-device symbol called a "breakout". It is not legal to attach a signal line directly to a bus line and any attempt to do so will elicit a warning box. In DesignWorks, a breakout is treated as a device with certain special properties. This means that it can be placed in any desired orientation, moved, duplicated, etc. using any of the device editing features available. A typical breakout appears as follows:



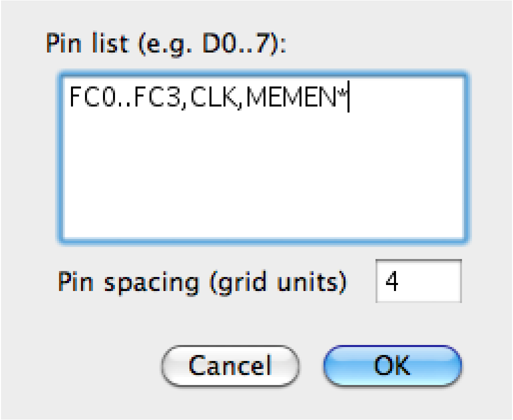
Any breakout can always be attached to any bus. When a breakout is attached that contains signals unknown in that bus, the signals are implicitly added to the bus. For example, if a second breakout was attached to the same bus containing only the signals CLK and CLR, all signals in all attached breakouts would be considered part of that bus. In this case, the list would be D0, D1, D2, D3, D4, D5, D6, D7, CLK and CLR.

Any combination of the internal signals can now be brought out of the bus at any point, as in this addition to the above circuit:



## Creating a Breakout

To create a breakout, select New Breakout command in the Options menu. If the new breakout is to be similar to an existing one, first select the similar breakout or the bus to which the new breakout is to be connected. This dialog box will appear:



Pin list (e.g. D0..7):

FC0..FC3,CLK,MEMEN*

Pin spacing (grid units) 4

If a bus or breakout was selected on the circuit diagram then the breakout info box will display a list of the signals in that bus or breakout, otherwise it will be empty. If this list already matches the signals you want in the new breakout, then just click the "OK" button or hit Enter on the keyboard. Otherwise, edit the signal list, noting these options:

blanks or commas can be used to separate individual names in this list, therefore bussed signals cannot have names containing a blank or comma.

a range of numbered signals can be specified using these formats:

`D0..7` or `D0..D7`

is equivalent to

D0 D1 D2 D3 D4 D5 D6 D7

D15..0

is equivalent to

D15 D14 D13 D12 D11 D10 D9 D8 ... D0

D15..D00

is equivalent to

D15 D14 D13 D12 D11 D10 D09 D08 D07 ... D00

Note that the ".." format implies that bussed signal names cannot contain periods.

the signals specified will always appear in the order given in this list from top to bottom in standard orientation. We recommend always specifying numbered signals from lowest numbered to highest, as in the first example above, since this matches the standard library symbols.

there is no fixed limit on the number of signals in a bus, but we recommend dividing busses up by function (i.e. address, data, control, etc.) for ease of editing.

any combination of randomly-named signals can be included in the list, as in these examples:

D0..15 AS* UDS* LDS*
CLK FC0..3 MEMOP BRQ0..2

Once the list has been entered, click on the OK button or hit the Enter key. An image of the breakout will now follow your mouse movements and can be placed and connected just like any other type of device.

### Setting Breakout Pin Spacing

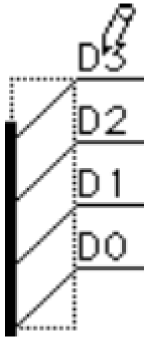
The number in the Pin Spacing box will be the spacing between signal pins on the breakout symbol, in grid units. The default value is 4 to match the standard DesignWorks libraries, but any number from 1 to 100 can be entered.

### Editing Breakout Pins

The signal name notation that appears on a breakout pin is actually a pin attribute. It can therefore be edited by the usual attribute editing mechanisms, i.e. either:

Select the pin and choose the Get Info command in the Options menu, then click the Attributes button, OR,

Click the text cursor directly in the text on the schematic, as follows:



Type the desired new name.

Press the Enter key. The breakout pin and the attached signal will be renamed as entered.

**IMPORTANT: The notation on the breakout pin is always the same as the name of the attached signal name. Changing the breakout pin renames the attached signal and will detach it from any like-named signals already in the bus.**

## Using Bus Pins

DesignWorks supports user-created bus pins on devices. A bus pin can be defined to have any collection of named internal pins. Note these properties of bus pins:

The bus pin itself does not represent a physical device pin. It is only a graphical place-holder on the schematic representing a group of internal pins. The bus pin itself never appears in a netlist.

The internal pins represent physical device pins. Even though they do not appear on the schematic, they can have all the same parameters as normal devices pins, including pin numbers and attributes. These parameters can be accessed using the Bus Pin Info command in the pin pop-up menu.

When a device with a bus pin is placed, it has a pre-created bus attached to it by default. This bus will contain one signal for each internal pin, with the initial name of the signal being the same as the name as the pin.

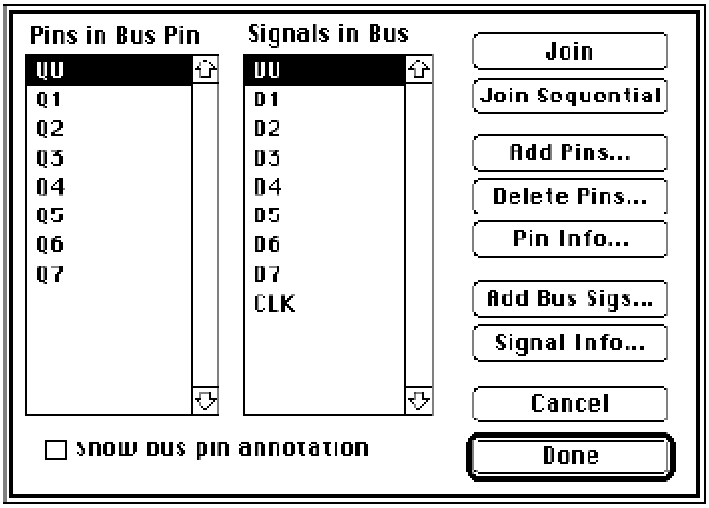
A "splicing" box can be displayed using the Bus Pin Info command in the pin pop-up menu. This box allows any internal pin to be connected to any signal in the attached bus.

***For more information on creating device symbols with bus pins, see "Placing a Bus Pin" on page 301.***

## Changing Bus Pin Connections

When a bus is connected to a bus pin on a device or hierarchical block, the bus internal pins will by default connect to signals with the same name in the bus. To change these default connections, use the Bus Pin Info command in the pin pop-up menu.

This menu item will be enabled only when a bus pin on a device is selected. It allows the association between the bus internal pins on the device and the signals in the bus to be changed. This box will be displayed:



The left-hand list shows the names of the pins contained in the selected bus pin. The right-hand list shows all the signals in the attached bus. For each pin in the pin list, the signal in the same row in the signal list is the one attached to it. Signals in the signal list beyond the end of the pin list are not connected in this bus pin. The following sections describe the operations available in this box.

### Changing Signal Connections in a Bus Pin

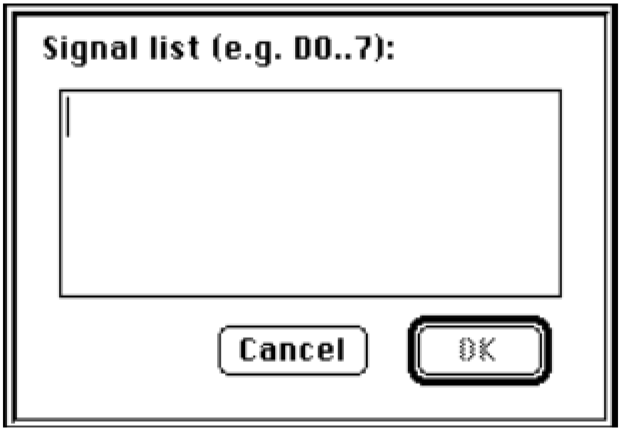
Two buttons are provided to change the association between pins and signals. The Join button causes the selected pin in the pin list to be joined to the selected signal in the signal list. If the selected signal is already attached to another pin in the list, then the signals will be swapped (i.e. a signal can only connect to one pin and vice versa). The signal list will be updated to show the new relationship.

The Join Sequential button provides a quick method of joining multiple numbered pins and signals. The selected pin is joined to the selected signal, as with Join, above. If the signal and pin names both have a numeric part, both numbers are incremented and the corresponding signal and pin are joined. This process is repeated until either the signal or pin name is not found in the list.

For example, given the lists appearing in the above picture, if pin D0 and signal D4 are selected, then Join Selected will join D0-D4, D1-D5, D2-D6 and D3-D7. Since there are no more numbered pins, the process would stop. Note that although the signal and pin names are the same in this example, this is not a requirement.

### Adding Signals to the Bus

The Add Bus Sigs button allows you to add signals to the signal list so that they can then be joined to device pins. Clicking this button displays this box:



A list of signals can be typed into this box using the same format as the New Breakout command. Following are examples of allowable formats:

```
D0..7
D0..15 AS* UDS* LDS*
CLK FC0..3 MEMOP BRQ0..2
```

The order of entry will affect the order that the signals appear in the list, but is otherwise not significant. For a complete description of the rules of this format, see the New Breakout command elsewhere in this chapter.

**NOTE: These signals are only added temporarily. When you close the Bus Pin Info box, all signals that are not connected to any pin are removed from the bus.**

### Adding Internal Pins to a Bus Pin

The Add Pins button displays a text entry box similar to that described under “Adding Signals to the Bus”, above. The text format options available are the same as for signals. The resulting list of pins is added to the definition of the bus pin.

**WARNING: This is changing the definition of the device symbol. This will affect all other devices of the same type in the design and will make the symbol different from its original library definition.**

Note these additional effects of adding pins:

When a pin is created, a signal with the same name is also implicitly created and the pin is attached to it.

If a signal with the same name already exists in the attached bus, the new pin will be attached to it by default, unless you explictly do a Join in the pin/signal list.

Adding pins affects all other devices of the same type, but the signal connections you make in this box only affect this one instance. If the default connection is not apprpriate, you need to manually visit the other instances of the same type and update their connections also.

### Deleting Internal Pins from a Bus Pin

The Delete Pins button removes all selected pins in the pin list from the definition of the device symbol.

**WARNING: This is changing the definition of the device symbol. This will affect all other devices of the same type in the design and will make the symbol different from its original library definition.**

### Getting Pin Information on Internal Pins

The Pin Info button brings up the standard Pin Info box for the pin selected in the pin list. See the Get Info command for more information.

### Getting Information on Signals in the Bus

The Signal Info button brings up the standard signal info box for the signal selected in the signal list. This box is described in “Getting and Setting Signal Information” on page 98.

### Displaying a Bus Pin Annotation

If this option is enabled, a list of the connections made in the bus pin will be displayed adjacent to the pin. This is done by creating a value in the BusInfo attribute field for the selected bus pin. This value can be edited manually, if desired, but will be updated automatically each time this box is displayed. The format of the signal list is the format used by the Add Bus Sigs option, above.

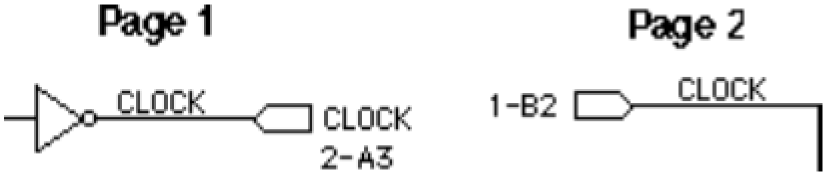
## Inter-page Connections

Signals drawn on different pages will initially have no logical connection, even if the names are the same (see the rules “Signal Connectivity Rules” on page 232). Inter-page connections are made using the Page Connector pseudo-device which is in the Pseudo Devs library. When attached to a signal line, the Page Connector makes the name of that signal known across all pages, i.e. any like-named signal on another page which also has a page connector will be logically connected to this one. The Page Connector can be placed anywhere along the signal line although it is normally convenient to place it at the edge of the page. The Page Connector itself can be named (in order to take advantage of the automatic page references mechanism described below), but it must have the same name as the attached signal. Changing the name of the Page Connector (using the naming procedure for devices) will also change the name of the attached signal. Likewise, changing the name of the signal will rename any attached Page Connectors.

### Automatic Display of Page References

DesignWorks has an Automatic Page Reference facility which displays the page number and position of any Page Connectors associated with a given signal. These page references are automatically displayed immediately adjacent to any Page Connector as soon as multiple Page Connectors are attached to the same signal.

For example, in this next case both pages have signals named CLOCK with Page Connectors attached, so the two CLOCK signals are logically connected. Note that both Page Connectors are notated with the page number and grid reference:



When a name is applied to the Page Connector itself, the page reference will appear under the name.

As long as automatic updating is enabled, these references will be updated automatically when any Page Connectors or attached signals are added, deleted, moved or renamed.

### Enabling and Disabling Automatic Page References

Automatic page reference updating is controlled by the Design Preferences command in the Drawing menu. Automatic references are enabled whenever the Automatic Page References box is checked. This means that page reference text will automatically be placed next to any named Page Connector that is placed in the design, and will be updated whenever any page connector is moved. If they are disabled, the current page reference settings will be left untouched when any schematic edting is done.

### Manually Updating Page References

If automatic page references are disabled, page references will be updated only when the Update Now button in the Design Preferences box is clicked. This can be used for large designs where page reference updating may cause delays while editing.

### Setting Page Reference Format

To change the page reference format, select the Design Preferences command in the Drawing menu. Two aspects of the page reference format can be controlled:

> The "Format" text item controls the appearance of each page reference in the reference list.
>
> The "Max. Width" item controls the number of references that will appear on each line before creating a new line. For signals with many connections, this prevents unwieldy page reference strings.

Three characters are special in the format string:

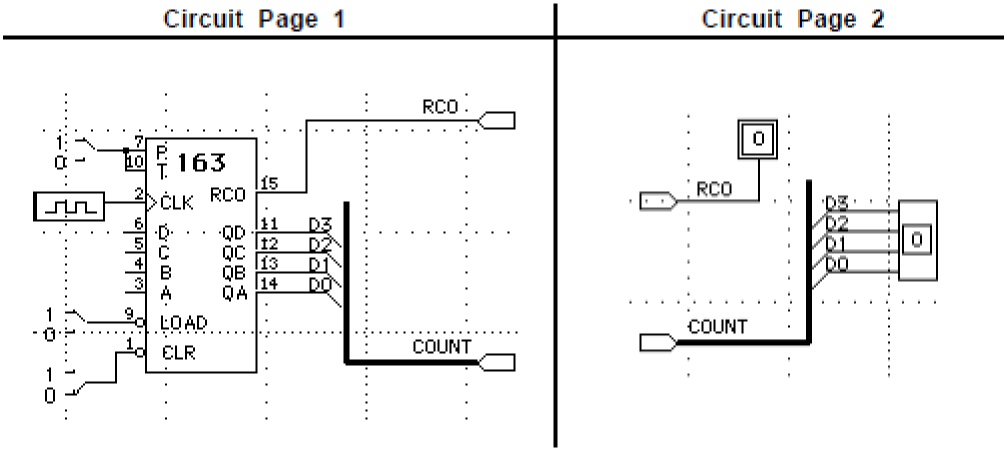
| | |
|---|---|
| **P** | will be replaced by the page number |
| **X** | will be replaced by the X grid position |
| **Y** | will be replaced by the Y grid position |

The default format is "P-XY" which generates pages references like "2-C4" (i.e. page 2, grid position C4). All characters other than the special ones above will be placed in the page reference list verbatim.

The maximum length of a format string is 16 characters.

## Connecting Busses Across Pages

Busses can also be connected between pages by the same method, except that the Bus Page Connector must be used in place of the Page Connector. This is illustrated in this simple circuit:

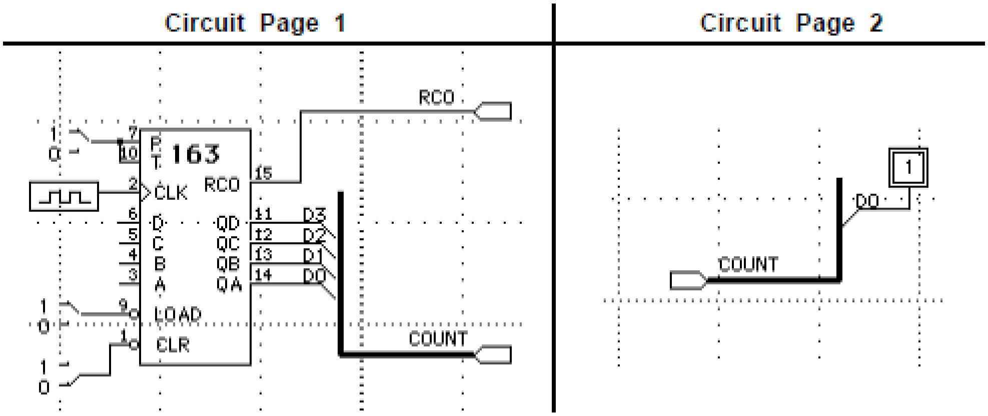


## Using Page Connectors on Internal Bus Signals

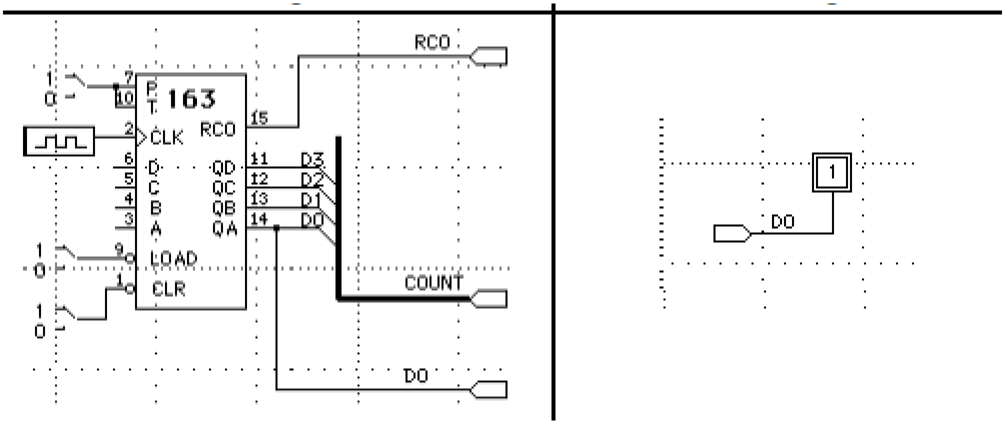
Note that a Bus Page Connector <u>does not</u> make the names of the internal signals known globally, only the bus itself. To bring an individual internal bus signal across to another page, one of the following methods can be used:

1) Make the entire bus global using a bus page connector, then use a

breakout to access the desired signal, as in this example:



2) Place signal page connectors on the individual signal on both pages, as shown here:



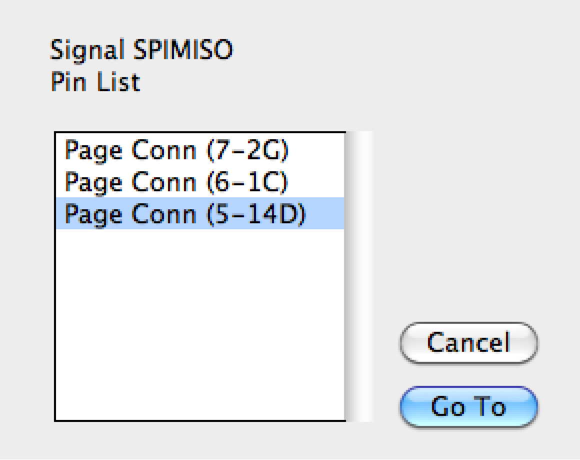
## Changing the Page Connector Symbol

Note that the symbol for the Page Connector device can be changed using the DevEditor tool just as with any other device. In order that the resulting symbol be recognized by the program as a Page Connector, you must either start with an existing Page Connector symbol, or set the primitive type setting appropriately.

***See "Creating a Page Connector" on page 319 for specific information on creating Page Connector symbols.***

## Tracing Connections Through Page Connectors

If a Page Connector device is selected in the schematic, the Get Info command displays this box:



Signal SPIMISO
Pin List

Page Conn (7-2G)
Page Conn (6-1C)
Page Conn (5-14D)

This is essentially the same box as is displayed for the signal Pin List command, except that only Page Connectors are listed. Following on each item is the page number and grid reference of the item.

To display the selected page connector, either:

select the item in the list and click the Go To button,

OR

double-click on the item in the list.

## Power and Ground Connections

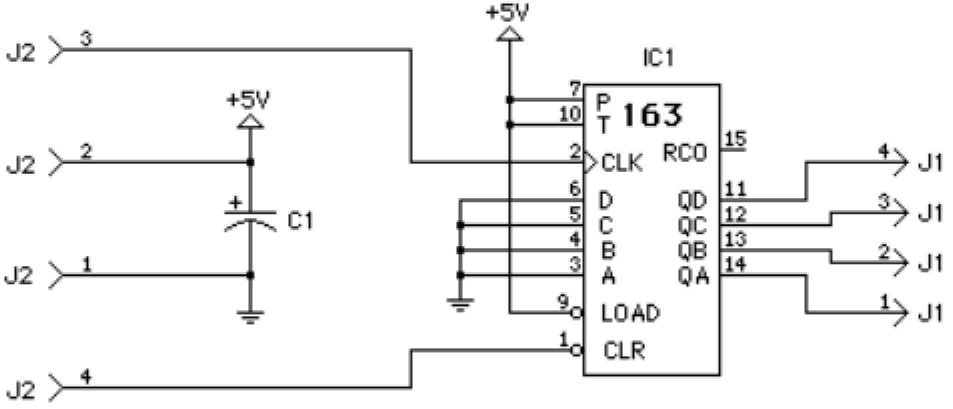
When a schematic diagram is created, the power and ground connections for devices are normally not drawn as signal connections on the diagram. These connections would clutter the diagram and are not necessary for an understanding of the logical function of the circuit. Obviously, though, these connections must be included at some level in order to form a complete netlist. Three methods are available to do this:

Power and ground pin connections can be specified as attributes for each device. The Schematic tool itself attaches no significance to the attribute fields, but the report generator reads them and adds the specified pins to the netlist. The libraries of standard types supplied with DesignWorks include this information for each type. It is only necessary for you to enter this data if you are making non-standard connections or using a device that has no default power connections. The pre-defined attribute fields "Power" and "Ground" are intended for this purpose.

Power and ground pins can be added to each device symbol (using the DevEditor tool) and the connections shown as normal signal lines on the diagram, or connected using Signal Connector (power and ground) symbols. The standard DesignWorks libraries are not set up in this fashion. This method is normally used only for unusual design situations such as an isolated supply for a specific part in a design.

A separate symbol can be created for each device showing only power and ground connections. This allows all power wiring to be shown in one part of the diagram or on a separate page.

In fact, these methods are not mutually exclusive and can be used in combination as needed. This will be illustrated by way of a simple example:



The attributes for the device IC1 were specified in the library entry for type "163" and are as follows:

| | |
|---|---|
| **Power** | 14 |
| **Ground** | 8 |

In the above diagram, the signal named "+5V", the symbols labeled "+5V" and the attribute entries "Power" will all be merged into a single list of connections in the netlist, as follows:

```
S00005IC1-14J1-1
S00007IC1-2J2-3
S00008IC1-1J2-4
S00009IC1-11J1-4
S00010IC1-12J1-3
S00011IC1-13J1-2
GroundC1-2IC1-3,4,5,6,8J2-1
+5VC1-1IC1-10,7,9,16J2-2
```

**NOTE: The standard report form files add the entries in the Power field to a net named "+5V".**

The demonstration files provided with DesignWorks provides more complex examples of this usage.

## Power and Ground Naming Convention

The standard libraries provided with DesignWorks include the fields Power and Ground for the standard power and ground pins. Since field names cannot contain special characters, we recommend using the words "Plus" and "Minus" to create attribute fields for other supplies. For example, pins to be attached to a -12V supply would be listed in field "Minus12V". The Report Generator allows the name of the signal net to be different from the field name, if desired.

The supply and ground signal connectors can be found in the Connectors library. The standard device libraries include entries for Power and Ground and other standard power pins. As in the above example, other pins can be attached to these lines using symbols placed on the diagram.

***See "Extracting Power and Ground Connections from Attributes" on page 367 for instructions on adding other power nets.***

## Power and Ground Connections in

## Attributes

DesignWorks allows any number of special attribute field names to be specified as "signal sources". When a report is generated, the Scripter searches the attributes attached to each device for fields with these names. Any pin numbers specified in the field will be attached to a list for the signal of the same name. E.g. the value "14" in the Power field will cause pin 14 on this device to be attached to the signal named as the power signal. Multiple pins can be specified using commas, e.g. "1,2,14".

**NOTE: Power and ground connections made through attributes do not appear as logical connections during schematic editing operations. The merging of these pins into nets occurs only when generating netlist output and exists only in the netlist. These connections will not appear in the Pin List command and will not be simulated by the DesignWorks Simulator option.**

**IMPORTANT: The use of signal sources in netlist output is completely dependent on the netlist generating script. DesignWorks does not perform this function automatically if it is not specified in the script. Whenever appropriate, the settings for the standard power and ground fields described here have been included with the standard report scripts provided with DesignWorks. However, this can be modified by users and depends on details of the destination system.**

You can find more information on the use of power and ground fields in the netlist format you are using in these locattions:

The ReadMe file provided with the Design Kit you are using.

The Format Notes built in to the netlist script. For more information on how to see these, see “Viewing Format Notes” on page 370.

“Extracting Power and Ground Connections from Attributes” on page 367 provides more information on specifying signal sources in a report script.

## Signal Connector (Power and Ground) Symbols

DesignWorks uses a type of pseudo-device symbol called a "signal connector" to maintain connectivity between like-named power and ground symbols that are used on circuit diagrams.

As soon as a Ground symbol is placed on the diagram, the attached signal will be named "Ground" (the name will be invisible initially). This will cause it to be connected by name to any other signals having Ground symbols, or explicitly named "Ground".

Connectivity can be checked at any time by double-clicking on any ground or power line. This will highlight all other like-named lines on the diagram.

**IMPORTANT: Signal connectors do not cause a logical connection to be made between hierarchy levels in a hierarchical design. These signals can be merged into a single net for netlisting purposes by specifying the net in question as a signal source. However, this processing is done as the report is generated and does not create a logical connection in the design, e.g. for interactive simulation.**

***See "Making Signal Connections Across Hierarchy Levels" on page 262 for more information on signal connections in hierarchical designs.***

### Using Signal Connector Devices

Signal Connector Devices are placed on the diagram just like any other DesignWorks device. A set of standard power-supply symbols are included with DesignWorks in the Connectors library. If you connect two different signal connector devices together, you will be prompted to provide a name for the resulting signal.

### Creating Signal Connectors in a Library

Signal Connector devices (like Page Connectors) are special primitive "pseudo-devices" in DesignWorks and can be created using the Set Primitive Type command in the DevEditor tool to select the SIGCONN primitive type.

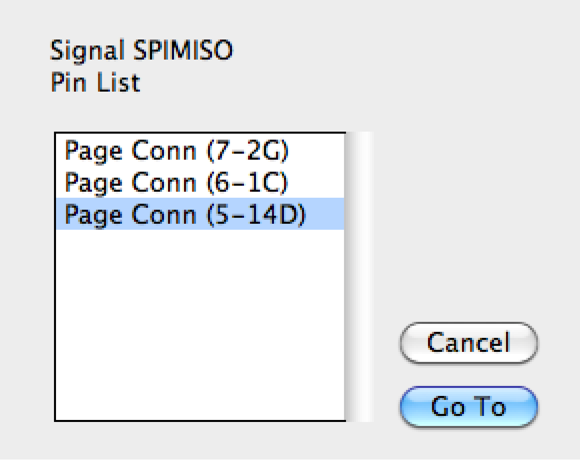
**NOTE: The signal attached to a signal connector device is actually named to match the pin name of the signal connector pin specified in DevEditor, not the type name. In**

**most of the power and ground symbols provided with DesignWorks, these two names are the same. However it is possible to create a symbol called, for example, "Ground" in the library which actually names the attached signal "GND". The Ground symbol in the SPICE Devices library provided is an example of this in that it names the attached signal "0" to match the SPICE ground naming convention.**

***See "Creating a Power (Signal) Connector" on page 318 for more detailed information on this procedure.***

### Tracing Connections Made by a Signal Connector

If a Signal Connector device is selected in the schematic, the Get Info command displays this box:



Signal SPIMISO
Pin List

Page Conn (7-2G)
Page Conn (6-1C)
Page Conn (5-14D)

This is the same box as is displayed for the signal Pin List command. Following on each item is the page number and grid reference of the item.

To display the selected pin, either:

Select the item in the list and click the Go To button,

OR

Double-click on the item in the list.

# Using Signal Auto-Naming

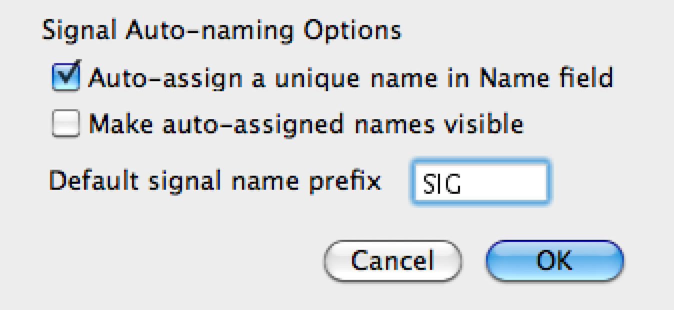
DesignWorks has an automatic name assignment feature that ensures that every signal object that is created has a distinct name. This ensures that it is always possible to track items in a netlist or error report back to a signal on the schematic. Signal auto-naming is enabled by default when a new design is created, unless the selected template has specifically disabled it.

When Auto-Naming is enabled, all signals will be given a default name when they are created. In addition, whenever an editing operation causes a single signal to be broken into two, or part of a signal to be copied without a visible name label, a new, unique name will be auto-assigned. This ensures that every object in a design has a unique name and can be identified in a netlist.

**NOTE: The program generates new names using a specified prefix and the signal's token number, which will produce a unique name in most cases. However, the program does not check for uniqueness. If you have manually assigned a name to another signal that matches the auto-assigned format, there is the possibility that an auto-assigned name may already exist. See "How Names are Generated" on page 230. To guarantee name uniqueness, you should run a Duplicate Signal Name error checking script. See "Using the ErrorScript Tool" on page 166 for more information on error checking.**

## Enabling Auto-Naming

Auto-Naming is enabled by selecting the Signal Naming Options command in the Naming and Packaging Options sub-menu in the Options menu. This will display the following options box:



Signal Auto-naming Options

☑ Auto-assign a unique name in Name field

☐ Make auto-assigned names visible

Default signal name prefix: SIG

Checking the "auto-assign" box causes all devices placed in the design subsequently to be given a default name if they are not already named.

If the "Visible" option is selected, the default name will be displayed on the diagram adjacent to the device or signal when it is created.

The "default signal name prefix" text box allows you to specify the characters to be used as a basis for auto-generated names. Changing this value does not affect existing names.

**NOTE: Enabling signal auto-naming does not assign names to existing signals that do not already have them. This must be done manually or using a script.**

## Disabling Signal Auto-naming

Signal auto-naming can be disabled by taking these steps:

Select the Signal Naming Options command in the Naming and Packaging Options sub-menu in the Options menu.

Turn off the Auto-assign box.

**WARNING: We do not recommend disabling auto-naming. When signal auto-naming is disabled, no modifications are made by the program to existing signal names. For example, if you split an existing signal into two parts, both parts will retain the existing name until you explicitly change it. This may result in an apparent short in a netlist.**

## How Names are Generated

The auto-generated name consists of two parts, the fixed prefix and the numeric suffix. The prefix portion is derived from the signal name prefix set using the Design Preferences command (which is stored in the design’s SigPrefix attribute field).

The numeric portion of the name is generated from the signal's "token" value. This value is guaranteed to be unique within a circuit level, but no attempt is made to fill in unused values. Thus, in a design that has been edited, sequential numbering is not guaranteed.

The program does not check names for uniqueness. If you have manually assigned a name to another signal that matches the auto-assigned format, there is the possibility that an auto-assigned name may already exist.

## Using Signal Token Values

Every time a device or signal is created in a DesignWorks circuit, it is assigned an integer value known as its "token". The token number stays with the device or signal for its lifetime and numbers are not re-used. This ensures that a given device or signal can always be recognized despite duplicate names or name changes. The token is used for a number of internal operations in DesignWorks, but can also be seen by the user in the following circumstances:

> The token number is used to generate default names for devices and signals, as described elsewhere is this chapter.
>
> The token number can be written out in netlists or bills of materials whenever a guaranteed-unique identifier is needed.

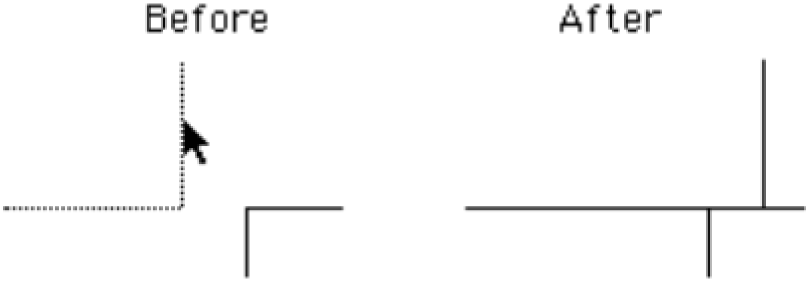
Note these characteristics of tokens:

> Tokens are assigned independently for each circuit in a hierarchical design and are thus only unique within a circuit, not across the entire design.
>
> When two signals are joined, one of the two tokens is discarded at random.
>
> Each logical symbol on the diagram (including pseudo-devices) has its own token. In a netlist, several symbols may be combined into a single package, so there is not necessarily a one-to-one correspondence between tokens and physical packages.

## Signal Connectivity Rules

This section provides a complete reference for all the methods used to interconnect signals. Each of these items is covered in more detail elsewhere in this manual.

DesignWorks automatically checks for a connection whenever the endpoint of a signal segment is moved. In most circumstances, any two lines that appear graphically connected are also logically connected. However, there are some editing operatons that can cause two lines to overlap without creating a logical connection, for example:



In this case, the corner of one signal overlaps another, but the free ends never touch. Since the program only checks free ends for connections, no connection is made between these items. This can be seen visually in this case by the lack of a connection dot at the intersection. In addition, clicking on either of the signals would highlight only the lines belonging to that signal.

Signal names must be visible to be checked for connections, unless a Signal Connector device (e.g. Ground) is attached. If you use an attribute editing command to make a formerly visible name invisible, any connections caused by that name will be broken.

Signal names are known throughout a single page. Like-named signal traces on a single page are thus logically connected for simulation and netlisting purposes. Whenever a signal name is added or changed, the circuit is checked for a change in connectivity. If the name is now the same as another signal on this page, the two signals are merged into one. If this signal segment was previously connected by name to others and the name is changed, then the logical connection is broken. Whenever a name change causes two signals to be connected, both parts will flash on the screen to confirm the connection.

Signal names are global across all pages of the circuit when a page connector symbol is added to the signal line. Thus if the name is changed on a signal line having a page connector then all circuit pages are checked for like-named signals having page connectors. If any such signals are found then a logical connection is made between them.

Signals which are contained in busses are a special case. All signals contained in busses have a name, even if this is not displayed on the diagram. However, the names of bussed signals will not be used to make logical connections unless an explicit name label has been added to the signal line.

For example, if you have a bus containing a signal named CLK and a separate signal line also named CLK, there will be no logical connection between these two signals. The name appearing on the bus breakout is part of the breakout symbol and is not considered to be a name label. If an explicit label is added to the bussed CLK signal (using the Name cursor) then the two CLKs will be logically connected.

The same rules discussed above for signals also apply to busses. Whenever two busses are logically connected, all like-named internal signals also become logically connected. Note that connecting busses across pages requires a "Bus Page Connector" device rather than a "Page Connector" to avoid compatibility problems between bus and signal connections.

# Hierarchical Design

# 11

This chapter provides background and detailed procedures for the hierarchical design features of DesignWorks.

## General Concepts

### What is Hierarchy?

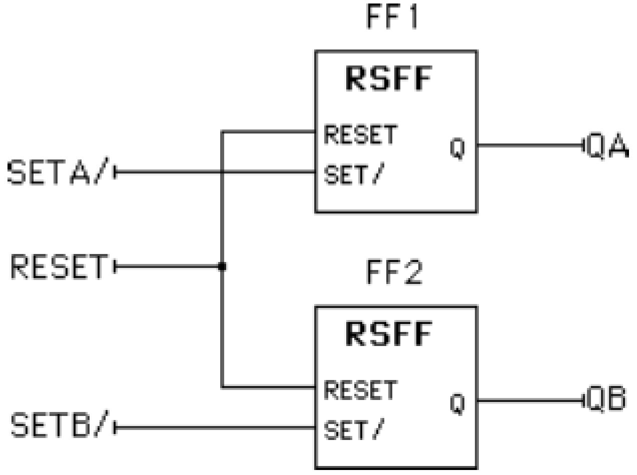
"Hierarchy" refers to the ability to have a "device" symbol in a schematic actually represent an arbitrary circuit block. The "pins" on the device symbol represent connections to specific input-output points on the internal circuit. For clarity, a device symbol that represents an internal circuit will be called a "hierarchical block".

Hierarchical design provides a powerful way of representing complex designs in compact and readable form. A top-level diagram of your system can show only major functional blocks. These blocks can then be opened to show more and more design detail.

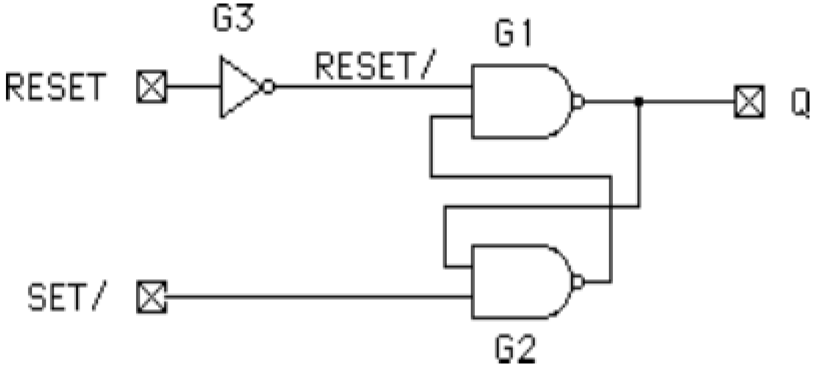
Hierarchical design in effect adds a "third dimension" to a schematic diagram. It also raises some complex issues that should be understood before embarking on a major design. Please review this chapter carefully before making extensive use of the hierarchical features of DesignWorks.

### A Simple Hierarchy Example

The following diagram is the master circuit, or top level, of our design example:



Note that it contains two symbols, both representing hierarchical blocks. Both symbols are of the same type, "RSFF", and therefore share the same internal circuit definition. The two blocks are named "FF1" and "FF2". Opening either one of these blocks reveals the following internal circuit:



This circuit consists of three device symbols, "G1", "G2", and "G3", representing physical devices, and a number of "port connector" symbols. The port connectors define the interface between the internal circuit and the pins on the symbol representing it.

Note the following characteristics of this simple design:

The block "RSFF" has been used twice, so there are actually two G1s, one inside FF1 and one inside FF2. We say that there are two *instances* of G1. Similarly for G2 and G3.

The signals "SET/", "RESET" and "Q" in the internal circuit will actually get absorbed into the attached signals in the parent circuit because they are attached to port connectors. They do not exist independently in the physical circuit.

Signal "RESET/" does not connect to a port connector, so it represents a separate signal in the internal circuit. Like the devices G1, etc., each signal in RSFF actually represents two physical signals.

These characteristics of a hierarchical design raise the following issues:

Device and signal names inside hierarchical blocks are not unique if the block has been used more than once. Therefore, a mechanism is needed to create a unique name for use in a netlist or bill of materials.

Signals attached to port connectors get absorbed into the signal attached to the parent symbol's pin. If this too is attached to a port connector, the process repeats upward until a unique signal is found. The name that will appear in the netlist will be the name of the highest-level signal.

A single symbol in the definition of an internal circuit can actually represent two or more physical devices. A mechanism must be provided to store separate information for each of these physical devices, e.g. for PCB placement or layout information.

When transferring data to PCB or simulation systems, it is usually necessary to produce a "flattened" netlist. I.e. a netlist representing the same circuit as if it had been expanded out on a single circuit level.

These issues are addressed in the following sections.

## Definition vs. Instance

In DesignWorks, circuit data for hierarchical blocks is separated into two groups:

Definition data is all information about the structure and connectivity of a circuit, plus the contents of all attributes not marked as "instance" fields. Definition data is associated with the parent symbol and will therefore be identical for all usages of the parent symbol.

Instance data refers to data kept with each physical instance of a device. For example, in the simple design used in the previous section, gate G1 will have two sets of instance data associated with it, one for each physical device that it represents. Instance data consists of pin numbers, interactive simulation data and any attribute fields marked as "instance" fields.

In a hierarchical design, only one set of definition data is kept for each hierarchical block type, whereas separate instance data is kept for each instance.

***More information on defining and using instance attribute fields is given in "Definition vs. Instance Fields" on page 174.***

## Choosing a Hierarchy Mode

DesignWorks implements a true hierarchical design structure. This means that "device" symbols appearing in a circuit can actually represent another, nested circuit. This subcircuit can be opened and edited at any time in separate circuit window. The term "design" is used in this manual to refer to a complete logical system, including the top level circuit and all subcircuits it contains.

DesignWorks has three hierarchy modes to address different design situations. These are described in the following sections.

### Flat Hierarchy Mode

"Flat" Mode is the simplest mode and is intended for most PCB-related designs or any smaller designs. Despite the name, it is still possible to create and edit internal circuits, but they are assumed to be for simulation or analysis only. Devices in subcircuits will not be assigned to physical packages by the Packager tool and will not appear in any netlist output.

### Physical Hierarchy Mode

Physical mode allows full use of hierarchical design, while still allowing data associated with individual device instances to be stored at any point in a design. It is intended for larger board-level or FPGA designs where interactive simulation is used or PCB layout information is to be kept with the design.

### Pure Hierarchy Mode

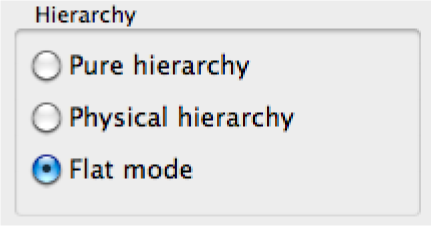
Pure mode stores only definition data for each device in a design. The "Keep With Instance" setting in attribute fields will be ignored. This allows very large designs to be represented in a small amount of memory, but does not allow interactive simulation or physical layout data to be associated with the design. It is intended for VLSI or other large-scale design applications.

## Setting the Hierarchy Mode

The choice of hierarchy mode is not carved in stone once made, since you can switch modes at any time. "Flat" and "Physical" modes are very similar and switching between them will normally only require that gate packaging be redone. "Pure" mode is substantially different, however, and should only be used after reading the discussion later in this manual.

**WARNING: Switching to Pure hierarchy mode from any other mode will cause all instance data to be lost. THIS CANNOT BE UNDONE!!! See the following section.**

To set the hierarchy mode for a design, select the Design Preferences command in the Drawing menu. The following controls allow selection of hierarchy mode:



Hierarchy

○ Pure hierarchy
○ Physical hierarchy
● Flat mode

Click on the desired button to select a new hierarchy mode, then click on the OK button. A warning box will ask you to confirm the mode change.

**WARNING: See the reference section earlier in the chapter for warnings about potential loss of data when switching hierarchy mode.**

## Effect of Changing Hierarchy Mode

The following table summarizes the effects of switching between hierarchy modes.

| | |
|---|---|
| **Flat -> Physical** | • No data is lost<br>• Attribute field usage for device packaging changes. Packaging will have to be redone |
| **Flat -> Pure** | • Any instance data associated with internal circuits will be lost |
| **Physical -> Flat** | • No data is lost<br>• Attribute field usage for device packaging changes. Packaging will have to be redone |
| **Physical -> Pure** | • Any instance data associated with internal circuits will be lost |
| **Pure -> Flat** | • No data is lost |
| **Pure -> Physical** | • No data is lost |

## Navigating in Hierarchical Designs

Hierarchy adds a third dimension to a design that requires some additional commands to allow you to move between levels. Here are some of the techniques available in DesignWorks for navigating around a hierarchical design:

> The Push Into command (or simply double-clicking on a subcircuit block) opens the subcircuit in a new circuit window. The Pop Up command performs the converse operation, closing the current subcircuit window and displaying its parent device symbol. These commands are described in the following sections.
>
> Any utility tool (like Browser, Find or ErrorScript) that can display a found object will open a subcircuit containing the object. These tools are described elsewhere in this manual.

### Opening (Pushing Into) a Subcircuit

The Push Into command opens the internal circuit of the given device in a separate window.

**NOTE: The Push Into command is available in the Options menu as well as in the device pop-up menu, i.e. by holding the Control key pressed while clicking on the device.**

In Physical Hierarchy mode, if you have used the same device type in multiple places in the design, the Push Into command creates a temporary type which is distinct from all other usages. When the subcircuit is closed you will be asked if you wish to update the other devices of the same type.

This menu item will be disabled (grayed out) under any of the following conditions:

The device is not a SUBCCT (subcircuit) primitive type

The device has its "restrict open" switch set in the Device Info box

If the selected device has no internal circuit, you will be asked whether you wish to create one. Clicking OK will create a circuit containing only the default port connectors matching the parent device.

Simply double-clicking on a device is a short-cut for the Push Into command.

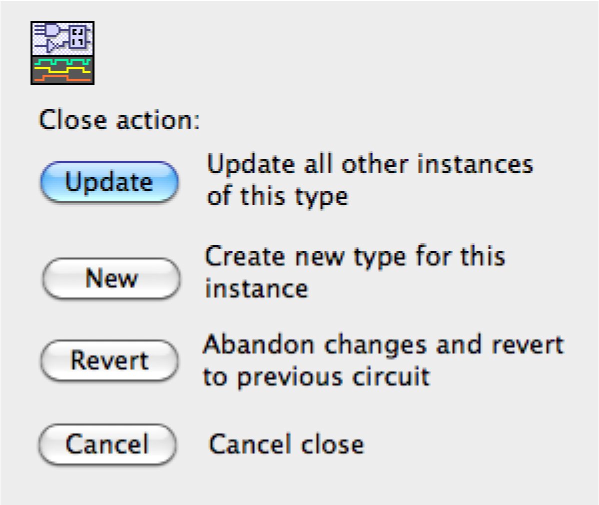
## Closing (Popping Out of) a Subcircuit

The Pop Up command closes the current subcircuit and displays the circuit page containing the parent device.

**TIP: The Pop Up command is also available in the circuit pop-up menu. That is, hold the Control key pressed while clicking anywhere in a circuit window not near any circuit objects.**

**NOTE: Clicking in the close box at the upper left corner of the window is equivalent to the Pop Up command, except that no attempt is made to display the parent device's window.**

If the design is in Pure hierarchy mode, or if the device being edited was the only one of its type, then the subcircuit will be closed without comment. If any changes have been made to the internal circuit that would affect other devices of the same type, then this box will be displayed:



Close action:

Update all other instances of this type

Create new type for this instance

Abandon changes and revert to previous circuit

Cancel close

The following close actions are available:

| | |
|---|---|
| **Update** | Clicking this button will cause the changes made to be copied to all other devices of the same type. This is the default action and can also be activated by typing the Enter key. |
| **New Type** | Clicking this button will make a new type definition for the device that was being edited. It will now be independent of any other devices that used to be the same type. |
| **Revert** | All changes made to the internal circuit will be discarded and it will revert to its old structure. |
| **Cancel** | Leaves the internal circuit open. |

## Locking and Unlocking Subcircuits

A subcircuit can be locked to prevent accidental opening. This is intended for cases where the subcircuit is derived from a library or is used only to implement a simulation model which should not be edited as part of the parent design.

To lock and unlock a subcircuit from the schematic:

Locate the subcircuit's parent device symbol in the schematic and select it.

Select the Get Info command in the Options menu.

Check or uncheck the Lock Opening Subcircuit box, as desired.

Click OK.

**TIP: The default locked status can be set when creating a hierarchical block symbol by setting the value of the Restrict attribute field while editing the symbol using the DevEditor tool. The allowable values for this field are given in Appendix A—Predefined Attribute Fields on page 373.**

## Creating a Hierarchical Block - Top Down

In concept, a hierarchical design can be created from the top down or the bottom up. In practice, most design processes probably use a mixture of the two methods, depending on the order in which system components were designed.

Working top down means creating the highest level of the design first and then working downward to more detailed levels. In effect, this always means that you create and use the symbol for a hierarchy block first, before you necessarily even know what the internals of the block will look like. Once you have completely created the higher-level circuit including the symbols for all the hierarchy blocks it uses, you then proceed to define the internal circuits of the blocks.

input/output pin defintion

To create a hierarchical block from the top down, the following steps must be followed:

1) Create the block symbol using DevEditor.

2) Place the symbol on the schematic.

3) Create the internal circuit by pushing into the new symbol.

These steps are described in the following sections.

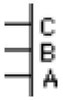
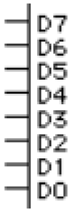
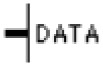
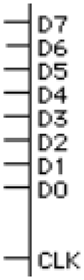
### Creating a Block Symbol

Symbols for hierarchical blocks are created using the DevEditor tool. DevEditor provides numerous tools and functions for symbol editing. We will provide only a quick summary of one technique.

1) Invoke the DevEditor by selecting its name in the Tools menu.

2) Select the Auto Create Symbol command in the DevEditor menu.

3) In the "Left Pin Names" text box enter a list of pins to appear on the left-hand side of the symbol. Following are some examples of pin lists:

| **Pin List** | **Pin Description** | **Symbol** |
|---|---|---|
| A B C | Three signal pins A, B and C |  |
| D0..7 | Eight signal pins D0, D1, D2, ... D7 |  |
| DATA[D0..7] | A bus pin called DATA containing 8 internal pins D0, D1, ... D7 |  |
| CLK,,,D0..7 | Nine pins in all with extra space between CLK and the others |  |

4) In the "Right Pin Names" text box enter a list of pins to appear on the

## Hierarchical Block Primitive Type

Hierarchical block symbols are simply device symbols which have the primitive type "SUBCCT", or subcircuit. Device symbols with any other primitive type cannot be used as hierarchical blocks. SUBCCT is the default primitive type when creating symbols with DevEditor, so it is normally not necessary to change this setting.

right-hand side of the symbol, using the same format as above.

5) Click the "Generate" button.

6) Make any desired graphical edits to the auto-generated symbol.

7) If you do not already have a suitable temporary library file open, create one now using the New Lib command (either in the File menu, or using the Parts palette pop-up menu).

8) Save the new part into the temporary library.

***See "Editing Device Symbols" on page 284 for more symbol editing techniques.***

## Placing the Block Symbol

Hierarchical block symbols are placed just like any other device symbol:

1) In the Parts palette, use the library selection menu to select the library that you saved the block symbol in.

2) Double-click on the name of the new symbol.

3) Place the symbol where desired on the schematic.

## Auto-Creating the Internal Circuit

In this section we will take advantage of the fact that DesignWorks creates a default internal circuit template when you open an empty block symbol.  It is also possible to create your own internal circuit from scratch.  This is discussed below in the section on "bottom-up" block creation.

Hold the Control key while clicking on the hierarchical block symbol.

Select the Push Into command.

A box will appear asking you to confirm that you wish to create a new internal circuit.  Click the OK button.

A new circuit window will now open with the default port connectors laid out according to these rules:

A default port connector symbol is placed for each pin on the parent block, placed according to their position on the symbol.

Each pin on each port connector has its type set according to the rules outlined in "Setting the Port Pin Type" on page 263.

# Creating a Hierarchical Block - Bottom Up

Any open design can be made into the subcircuit of a hierarchical block symbol. To create the internal circuit first and work upwards, follow these steps:

Create or open the design file that is to become the internal circuit and define its port interface.

Create and place the hierarchical block symbol in the target design using the Place Subcircuit command.

For this simple procedure we will make use of the Place Subcircuit command which performs a number of automatic operations for you. It is also possible to do some of these steps manually to get more control over the symbol graphics and port interface. These two extra steps are also described in the following sections:

Creating the hierarchical block symbol using the DevEditor tool is described in "Creating a Symbol for an Existing Subcircuit" on page 256.

Linking the subcircuit to the symbol using the Attach Internal command is described in "Attaching an Internal Subcircuit" on page 254.

**NOTE: This procedure links the subcircuit to the symbol only for this design. You can also store the subcircuit permanently with the symbol in the library. This allows it to be easily used in multiple designs without having to use the Attach Internal command each time. See "Creating a Part With a Subcircuit" on page 309 for details on how this is done.**

## Creating a Subcircuit

Creating a circuit that is to be attached to a parent symbol is exactly the same as creating an independent design, except for the necessity of defining the port interface. Note the following issues when creating the subcircuit:

Each signal that is to be an input/output point for the subcircuit (that is, a pin on the parent symbol) must have an attached port connector.

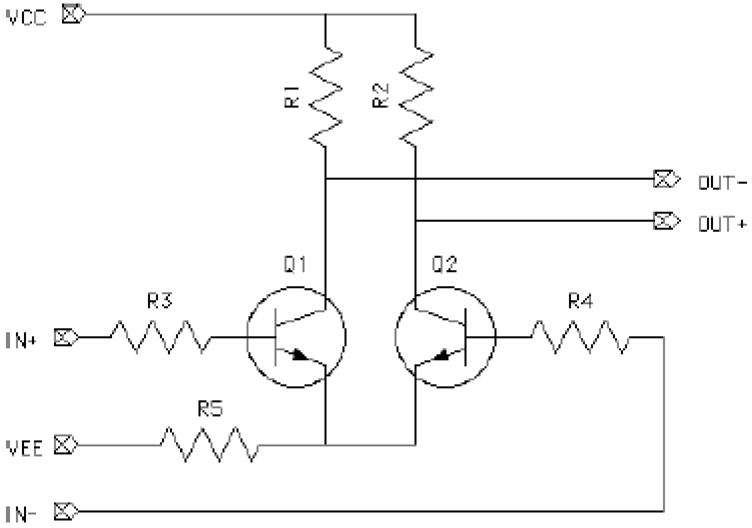
The name of the port connector device must exactly match the name of the associated pin on the parent symbol.

For each port, choose a port connector type (input, output or bidirectional) appropriate to the signal flow in your circuit. This allows for more efficient simulation, improved error checking and design clarity.

Bus ports require special care in creation and naming of port connector symbols. Review the rules given earlier in this chapter carefully.

***See "Creating and Using Port Connectors" on page 263 for complete details.***

The following sample shows how port connectors are attached to the connection points in a simple circuit:



**NOTE: You may wish to save this circuit as a design file for safekeeping, but this file will not be associated with the target hierarchical design in any way. The Place Subcircuit command described next will completely incorporate the subcircuit data into the target design with no reference to any external files. More information on alternate ways of**

**storing subcircuits is given in "Associating a Subcircuit with a Device Symbol" on page 252.**

## Placing a Subcircuit

In this section we will use the Place Subcircuit command to auto-generate a symbol corresponding to the subcircuit created above, and then place it in the target design. To do this:

Open or create both the subcircuit and the target circuit.

Bring to the front the target circuit window into which you wish to place the hierarchical block.

Select the Place Subcircuit command in the Subcircuit sub-menu of the Options menu. This will display a list of the open designs, except the topmost one, which is assumed to be the target.

Choose the circuit that is to become the subcircuit.

Click the OK button. Depending on the size of the subcircuit and the complexity of the symbol required, this operation may take a while. When the processing is complete, the cursor will be replaced by an image of the new hierarchical block.

Place the new symbol in the desired position in the target circuit.

# Generating Netlists from Hierarchical Designs

When producing a netlist for use by an external system, it is necessary to determine whether a "hierarchical" or "flattened" netlist is required.

## Generating Hierarchical Netlists

A hierarchical netlist retains the hierarchical structure of the original design. It contains only a single description of each type of hierarchical block used in the design. Normally, the lowest-level blocks are written out first, followed by higher blocks that refer to them, followed finally by the master circuit for the design. By its nature, a hierarchical netlist can contain only definition data.

This format is used by many simulators and FPGA packages, for example, SPICE.

Generally, any package that accepts a hierarchical netlist can also accept a flattened one. A flattened netlist may be useful in cases where unique instance data was stored with devices or signals in internal circuits.

## Generating Flattened Netlists

A flattened netlist has had all information about hierarchical structure removed from it. Every device and signal instance is listed separately with its instance data. Flattened netlists are normally required for PCB formats since name, unit and pin number assignments must be different for each device instance.

***See more information on hierarchical netlists in the entry Script Hierarchy Issues in the DesignWorks Script Language Reference (separate manual on disk).***

## Using Hierarchical Names

When producing a flattened netlist from a hierarchical design, it is usually necessary to generate a unique name for each device and signal in the design. This can be done in one of two ways:

> Manually or automatically assign a unique identifier to each item and store it in an "instance" attribute field. For example, in physical hierarchy mode, the Packager assigns a unique name to each device instance and stores it in the InstName field.
>
> Generate a unique identifier consisting of the definition name (i.e. Name attribute field) of the device or signal, prefixed by the names of all parent devices. For example, in the simple design example given at earlier in this chapter, the six physical devices would be called:

```
FF1/G1
FF1/G2
FF1/G3
FF2/G1
FF2/G2
FF2/G3
```

Assuming the Name field is unique within each circuit level, this is guaranteed to produce a unique identifier for each instance.

The "/" slash character is the standard name separator used in DesignWorks but can be changed by the user if needed using the procedure described below.

## Changing the Hierarchical Name Separator

The separator character used to generate hierarchical names is stored in the design attribute field HierNameSep. To change it:

1) Select the Set Design Attributes command in the Options menu.
2) Select the HierNameSep field in the field list.
3) Enter a new value in the data box.
4) Click OK.

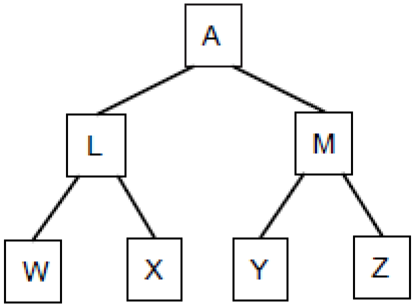
**NOTE: The usage of the hierarchy name separator and its appearance in netlist output is completely under the control of the netlist generation script. The script can override the format specified here. See the entry Script Hierarchy Issues in the DesignWorks Script Language Reference (separate manual on disk) for more information.**

# Printing Hierarchical Designs

A hierarchical design has an extra dimension which must be taken into account in determining the order of page printing.

## Determining Print Page Order

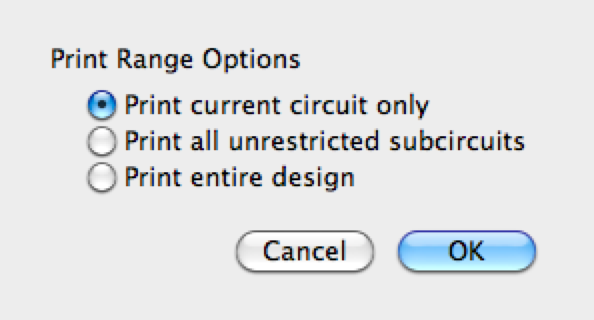
When a design is printed, the master circuit of the design is printed first in its entirety. Then, each hierarchical block is printed, followed by all hierarchical blocks it contains. For example, given the following structure of nested hierarchical blocks:



The print order would be: A, L, W, X, M, Y, Z. The order of printing of hierarchical blocks inside a single circuit is determined only by internal storage order and cannot be controlled by the user. E.g. W and X could be reversed above.

## Setting Printing Scope

The Print Range command in the File menu is used to set the scope of printing in a hierarchical design.



Print Range Options

● Print current circuit only
○ Print all unrestricted subcircuits
○ Print entire design

Three options are available:

Print current circuit only: With this setting, the Print Design command will print only pages belonging to the circuit level in the current window. With this setting, the page numbers shown by the "Show Printed Page Breaks" option can be used to select a subset of pages to print.

Print all unrestricted circuits: With this setting, the design's master circuit and all unlocked internal circuits will be printed. (The "locked" setting is controlled by a check box in the Get Info command for the parent device.) With this setting, the page numbers shown by the "Show Printed Page Breaks" option will not be relevant since they are numbered separately for each internal circuit.

Print entire design: Print the design's master circuit and all internal circuits, regardless of "locked" status.

## Printing Sequential Page Numbers in a Hierarchical Design

When a hierarchical design is printed, the $PRINTPAGENUMBER text variable can be used to apply sequential page numbers to the printed sheets.

**NOTE: This text variable only applies during a Print operation. When it is drawn on the screen, it is interpreted the same way as the PAGENUMBER variable, i.e. it gives the page number within the circuit.**

***See more information on text variables in "Using Text Variables" on page 110.***

## Associating a Subcircuit with a Device Symbol

There are a number of ways that a subcircuit can become associated with its parent device in a hierarchical design:

The subcircuit can be stored with the symbol in the library. This is referred to in DesignWorks as an "internal subcircuit". See "Creating a Part With a Subcircuit" on page 309 for information on using the DevEditor to attach a subcircuit to a symbol. Also see "Copying Symbols from a Design to a Library" on page 275 for information on using the Save to Lib command to store a symbol from a schematic to a library, including its subcircuit.

The symbol can be stored in a library with no associated subcircuit and one can be created or attached to it interactively after it has been placed in the schematic. This procedure is described in "Creating a Hierarchical Block - Top Down" on page 243.

The name and directory location of a circuit file can be stored with the symbol or added to the symbol after it is used in the schematic. The subcircuit itself is not stored with the symbol, just the information required to locate it later. Schematic options allow the subcircuit to be loaded automatically when the symbol is used, or on request at a later time. This is referred to as an *external subcircuit.*

**NOTE: Despite the name, the data for an "external" subcircuit is frequently loaded into memory and made part of the parent design. Generating reports and simulating using the DesignWorks Simulator option require that the external circuit be loaded into memory.** External subcircuit operations are described in "Working with External Subcircuits" on page 258.

**WARNING: When you attach or detach device subcircuits, you are in effect changing the definition of the device. This always affects <u>all other devices of the same type</u> in the same design. This applies to all the subcircuit commands following and any other subcircuit editing operation in DesignWorks. If you want to affect only a single device, you must first use the Make Unique Type command in the Part Type sub-menu to isolate it from others of the same type. See a more complete description of this command in "Making a Single Device Into a Unique Type" on page 278.**

## Working with Internal Subcircuits

The commands described in this section are intended to assist in assembling separate circuits into a complete, hierarchical design, changing the organization of a design, and moving or copying subcircuits between designs. See "External vs. Internal Subcircuits" on page 253 for more information on the usage of internal vs. external subcircuits.

### Placing a Subcircuit Block in a Parent Circuit

The Place Subcircuit command is a shortcut for creating a symbol to rep-

### External vs. Internal Subcircuits

The primary distinction between external and internal subcircuits is whether an association is maintained with another circuit file on disk:

> An internal subcircuit does not retain any information about its original source. The subcircuit is considered to be a permanent part of the parent design and no attempt is made to keep it up to date with any original file it may have been derived from. Use an internal subcircuit when the subcircuit forms a unique and essential part of this particular design, i.e. it is not a fixed block that is shared among many designs.
>
> An external subcircuit retains an association with a specific design file on disk. The subcircuit data may be loaded into the parent design for report generation or simulation, but the master copy of the data is considered to reside in the external file. Use an external subcircuit when the subcircuit is essentially a fixed "library" component that does not need to be edited uniquely for this design and might be shared among many designs. There are a number of menu commands specifically intended to assist in working with external subcircuits. These are described in "Working with External Subcircuits" on page 258.

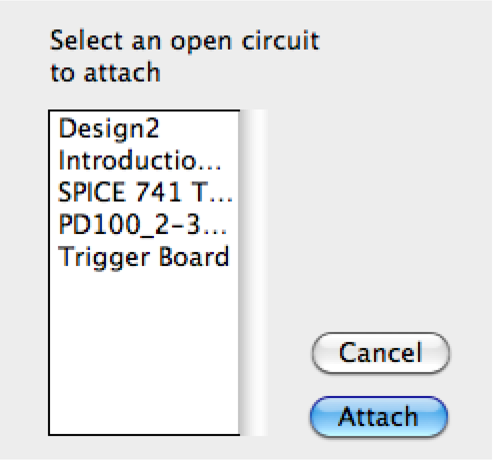
resent an internal subcircuit and placing it in a higher-level circuit. This command performs these steps:

> Displays a selection box allowing you select any other open design to become a new subcircuit block.
>
> Invokes the DevEditor's Auto Create Symbol function to generate a rectangular symbol with pins representing the ports defined in the subcircuit.
>
> Attachs the selected design as a subcircuit of the new symbol.
>
> Enters device placement mode, allowing you to place the new symbol in the current circuit.

Note that the generated symbol is not saved to a library, so if you cancel the device placement mode, you can only recover the symbol by going through the same process. If you wish to modify the auto-generated symbol after it is used, this can be done using the Edit Symbol command described in "Editing a Device Symbol in a Schematic" on page 283.

### Attaching an Internal Subcircuit

The Attach Subcircuit command in the Subcircuit menu allows you to select an open design to attach as a subcircuit to the selected device. When this command is selected, the following box will appear:



**WARNING: This operation cannot be Undone!!!**

Clicking Attach on the Attach Internal box will cause the following actions to be taken:

If the current design (i.e. the one containing the parent device) contains other devices of the same type as the selected device, then a separate, temporary type will be created for the selected device. This allows the definition of this type to be temporarily modified by the addition of the subcircuit. The other devices of the same type will be updated when you close the attached subcircuit.

The attribute definition table in the selected internal circuit is compared with the table in the master design. If the Attach operation would result in new fields being defined in the master design's table, then you will be prompted for confirmation. If you click OK, then new fields are merged into the master design's table.

Attribute values associated with the internal circuit are merged into the master design. If a given field already has a value in the master design, the value in the internal circuit is discarded.

The logical linkage between the selected device and the new internal circuit is completed. If any mismatch is detected between the port connectors defined in the internal circuit and the pins on the parent device, you will be warned.

The title of the internal circuit is updated to reflect its hierarchical position in the master design.

Auto-packaging, if enabled, is disabled since this operation potentially invalidates the design's package table.

The newly-attached internal circuit's window is brought to the front. It is now considered to be an internal circuit that has been opened for editing and modified. When you close the internal circuit, you will be asked if you wish to update other devices of the same type.

## Detaching an Internal Subcircuit

The Detach Subcircuit command makes the currently displayed subcircuit into a separate design and redefines the parent device as having no internal circuit.

**WARNING: This operation permanently removes the subcircuit from the selected device and all other devices of the same type in the selected design. If you do not wish to update other devices in the design, use the Make Unique command to isolate the selected device first.**

**The Detach operation cannot be Undone!!!**

In particular, Detach Internal performs the following operations on the subcircuit displayed in the frontmost window:

The circuit is unlinked from its parent device, making it into a separate design.

The title of the subcircuit is set to a default "Designxxx" name.

The internal circuits of all other devices of the same type in the design are removed.

Auto-packaging, if enabled, is disabled since this operation potentially invalidates the design's package table.

## Discarding a Subcircuit

To discard the subcircuit associated with a parent device:

Locate and select the parent device in its circuit.

Select the Discard Subcircuit command in the Subcircuit menu.

**WARNING: All data associated with the subcircuit is destroyed by this command. This cannot be undone!**

The Discard Subcircuit command redefines the selected device (and all others of the same type) as having no internal circuit.

## Creating a Symbol for an Existing Subcircuit

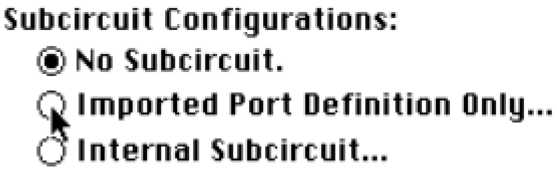
To create a symbol for a subcircuit that has already been created an is open in a DesignWorks window:

1) Select the DevEditor tool in the Tools menu.

2) Select the Subcircuit & Part Type command in the DevEdit menu.

3) Click on the Imported Port Definition Only button. This will display a

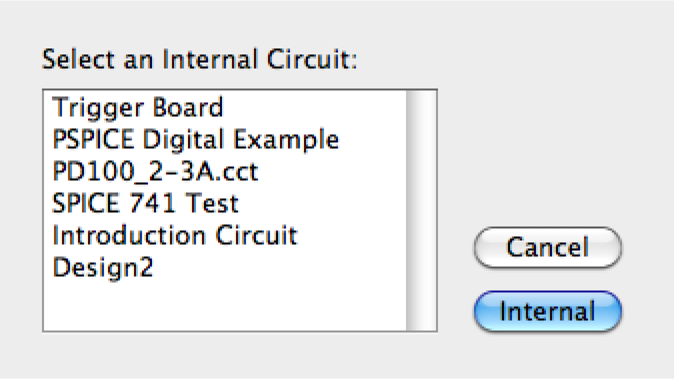
list of all currently open designs.



Subcircuit Configurations:
- ● No Subcircuit.
- ○ Imported Port Definition Only...
- ○ Internal Subcircuit...

**NOTE: Choosing the Internal Subcircuit option instead will cause the circuit definition to be saved in the library with the symbol.**

4) Select the desired internal circuit in the list and click the Internal button.



Select an Internal Circuit:

Trigger Board
PSPICE Digital Example
PD100_2-3A.cct
SPICE 741 Test
Introduction Circuit
Design2

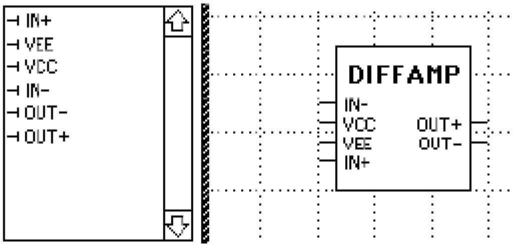
DevEditor now examines the selected circuit and creates a list of pins matching the port connectors in the circuit.

5) Select the Auto Create Symbol command in the DevEdit menu.

6) Click on the Extract Pin List button.

This will place input and output pins in default positions in the Left and Right Pin Names boxes. You may move names between these boxes as desired to determine placement of pins. Be careful not to add or delete any names. You can also change the part name that will appear in the symbol, if desired.

7) Click on the Generate button.  A default symbol will now appear in the DevEditor window.



This symbol can be edited manually using the drawing tools at the left.

8) Create a temporary library to save the symbol in.  This is done using the New Lib command in the Libraries sub-menu of the File menu, or in the Parts palette pop-up menu.

9) Select the Save Part As command in the File menu.  Select a suitable part name and double-click on the new library that was created in the last step.

10) Close the DevEditor window.

### Saving a Symbol with Internal Subcircuit to a Library

If you have created a hierarchy block symbol and its associated definition in a design and wish to save it in a library for use in other designs, follow these steps:

Create or open the desired destination library.

Locate and select the parent device symbol in your design.

Select the Save to Lib command in the Part Type sub-menu.

In the Save to Lib box, turn on the "Internal circuit" checkbox to indicate that the internal circuit should be saved in the library.

Click the Save button.

## Working with External Subcircuits

An external subcircuit is one which retains its association with an external circuit file, allowing it to be updated or reloaded as the external file is changed.  This mechanism was implemented specifically to load models for use with the DesignWorks Simulator, although it is valuable in any case where you wish to have a library of shared subcircuit definitions that can be loaded on demand. See "External vs. Internal Subcircuits" on page 253 for more information on the distinction between internal and external subcircuits.

**NOTE: In order to simulate, analyze or generate reports from a hierarchical design, DesignWorks requires that all subcircuits be loaded into memory. These operations do not automatically search for subcircuit data in external files. It is your responsibility to ensure that the loaded subcircuits are up to date before performing any operation that requires them.**

### Adding External References

The information used to locate an external subcircuit is called an "external reference" and is stored in a number of attribute fields associated with the device. External references can be added to a device in various ways:

When creating or editing a symbol in the DevEditor tool, an external reference is added by using the Subcircuit and Part Type menu command and selecting the External Subcircuit option. This allows you to locate the external file, load the port definition and associate device pins with port connectors in the subcircuit.

If the Schematic tool is being used in conjunction with the DesignWorks Simulator, the SimLoad tool can be used to automatically locate and set external references for standard library parts as they are used or after they have been placed in a schematic. More information on this mechanism is provided in the Simulator manual.

The Attach External Subcircuit command in the Schematic tool can be used to associate an external subcircuit with a symbol after it has been placed in a schematic.

The external reference attributes can be entered manually. THIS IS NOT RECOMMENDED because any slight error in names or entry format will result in the file not being located.

### Creating External Subcircuits

External subcircuits are simply normal DesignWorks design files with port connectors added to match the pins on the parent device. Following is a brief summary of the considerations affecting subcircuits:

In order to be loaded automatically using the SimLoad tool, the file must be named according to the conventions outlined in the DesignWorks Simulator manual.

A port connector should be placed in the subcircuit for each pin on the parent symbol, including no-connect pins. Failure to do this will result in the Schematic tool reporting an error each time the subcircuit is opened and closed.

Each port connector must be named to match the corresponding pin name on the parent device.

The type of each port connector should match the type of the corresponding pin on the parent device. E.g. An input pin on the parent device should match with a "Port Conn In", or other appropriate type.

The subcircuit can contain any combination of other devices, including nested subcircuits to any depth. The only restriction is that the subcircuit may not contain the parent type or any other type that would create a recursive nesting of blocks.

***Other considerations relating to the creation of subcircuits for simulation are discussed in Chapter 4 - Creating a Schematic for Simulation in the DesignWorks Simulator manual.***

## External Reference Attribute Fields

The reference to an external file is stored in a number of attribute fields associated with the device, collectively called an “external reference”. Manual editing of these fields is not recommended, since any slight spelling or format error will result in the external file not being loaded. The required fields are summarized here:

| | |
|---|---|
| **ExtCctName** | This field contains the name of the design file containing the external subcircuit. |
| **ExtCctPath** | This field contains the "path name" of nested directories leading to the file. The first item in the string is always the disk name, followed by a colon, followed by the names of nested folders, separated by colons. |
| **ExtCctDate** | This field contains an integer value expressed in decimal representing 1/60's of a second since Jan. 1, 1904. If this field is empty, the reference will be considered to be out of date. |

### Attaching an External Subcircuit

This new command allows you to associate an external design file on disk with a device symbol in a schematic. It operates only on a single selected device at a time, but always affects all other devices of the same type in the design. It will optionally set the external reference attributes only, or set the external reference and immediately load and attach the subcircuit.

The Attach External Subcircuit command is also a convenient way of checking the status of an external reference. It locates the external file and compares the file's modified date with the parent device symbol. The "up to date" status and any problems locating the file will be displayed in the status box. If the selected device has an external reference which is currently out of date, clicking the OK button will reload it.

### Update External Subcircuits

This command allows you to scan all or part of a design and load or update external subcircuits. This affects only devices that already have an external reference, created by one of the methods described previously. If the external file cannot be located, or if the external file is older or the same age as the last update, nothing is done. A summary box will be displayed after execution is complete showing counts of loaded and unlocated files.

**NOTE: This command can dramatically increase the memory space occupied by your design.**

### Clear External Subcircuits

This command allows you to scan all or part of a design and delete the subcircuits from all devices having an external reference. This affects only devices that already have an external reference, created by one of the methods described previously. Devices with subcircuit blocks that were not loaded using an external reference will not be affected.

This command is intended primarily to unload simulation models that were loaded using the SimLoad tool after they are no longer required. This can save considerable storage space for the design.

**WARNING: This command discards a lot of information and cannot be Undone!**

### Auto-load External Subcircuits

This command sets or clears the "Auto Load" option. It is disabled by default. When this option is enabled, each part is checked for an external reference when it is selected from a library. If there is a reference and the file can be located, it is loaded and attached immediately. Enabling this option has no effect on parts already placed in the design.

**NOTE: If you select a part in the Parts palette that has already been used elsewhere in the design, the existing definition is used and no subcircuit will be loaded.**

## Making Signal Connections Across Hierarchy Levels

Signal connections between the levels of a hierarchical design are made using one of these three techniques:

Port connections - This is the normal method of making connections across hierarchy levels. A pin on the parent device symbol matches with a port connector in the internal circuit

Signal connectors - This method makes use of “signal connector” pseudo-device symbols on the schematic and is intended for making power and ground connections. See “Making Power and Ground Connections Across Hierarchy Levels” on page 268 for more information on using signal connectors in hierarchical designs.

Signal sources - This method uses device attributes to specify global connections and is also intended for making power and ground connections. See “Making Power and Ground Connections Across Hierarchy Levels” on page 268 for more information on this method.

**IMPORTANT: The port connector method is the only one of these three that makes an iogical connection between hierarchy levels as soon as the object is placed in the schematic. The signal connector and signal source methods rely on the Scripter to make name associations when a netlist is generated. This has an important implication for users of the DesignWorks Simulator option: The Simulator does not recognize these types of connections for simulation**

**purposes. To make a signal connection between hierarchy levels for use with the interactive simulator you must use port connections.**

## Creating and Using Port Connectors

Port connectors are special pseudo-device symbols that associate a pin on a hierarhical block symbol with a signal in an internal circuit. Pre-defined port connector symbols for the three most commonly used pin functions are provided in the Pseudo Devs library with DesignWorks: Port In (Input), Port Out (Output) and Port Bidir (Bidirectional). For cases involving bus pins or any requirement for special symbol graphics, this section describes how to create custom port connecter symbols.

**NOTE: If you are working top down in a hierarchical design, you should rarely have to make your own port connector symbols. See "Creating a Hierarchical Block - Top Down" on page 243 for more information. If you are working bottom up, then you will need to define the port interface for a circuit using port connectors before making the parent symbol. See "Creating a Hierarchical Block - Bottom Up" on page 246 for a higher level view of this procedure.**

***The procedure for creating port connector symbol graphics is described in "Creating a Port Connector" on page 319.***

### Setting the Port Pin Type

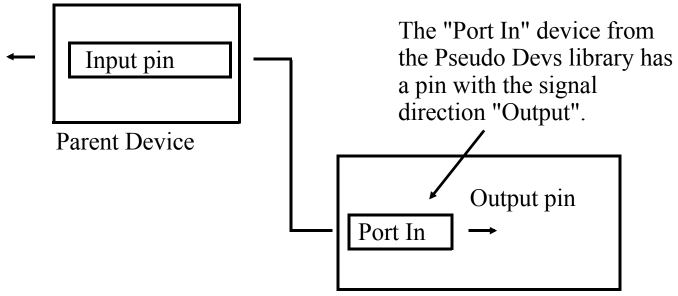
If pin type (input, output, bidirectional, etc.) is significant for your design, then the following rules should be noted when creating and using Port Connectors:

The pin on the Port Connector symbol is normally of the opposite type to the corresponding pin on the parent device symbol. E.g. A signal coming in to the hierarchical block is actually an output from the port connector pin.

DesignWorks does not check correctness of pin types on port connectors.

When an internal circuit is auto-generated, port connectors of a type opposite the parent pin are placed automatically.

This figure illustrates these points:



The "Port In" device from the Pseudo Devs library has a pin with the signal direction "Output".

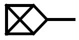
Internal Circuit

If the parent device pin is an output then it must be driven by the internal circuit, the internal circuit's port connector should have an input pin. To connect a signal in an internal circuit with an input pin on a parent device use the "Port Out" connector in the "Pseudo Devs" library.

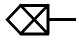
If the parent device pin is Bidirectional then the internal circuit's port connector should have a bidirectional pin as well.

**NOTE: When the *pin function* on a port connector is examined from the schematic module via the *Pin Info* dialog, the function shown is that of the parent type pin. This means that if the parent type pin is an input then the matching internal circuit port connector's pin will show as an input pin in the pin info dialog. If you examine the port connector in the DevEditor's pin palette, the same pin will show as an output.**

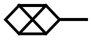
For normal signals (i.e. not buses) you can simply use one of the Port devices from Pseudo Devs library: Port Bidir, Port In or Port Out. These need to be named to match the pin names of the parent device.



Port In



Port Out



Port Bidir

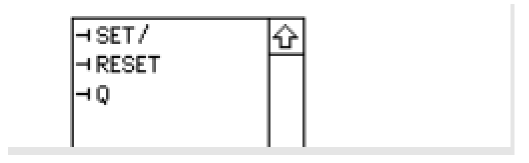
For buses, you need to create a custom port connector that lists all the signals to be carried across the interface. This procedure is described in the following sections.

## Creating Bus Ports

Connections can be made between busses across hierarchy levels using Bus Port Connectors. Bus pins on a parent device symbol must be matched with a Bus Port Connector having identical internal pins. For this reason, Bus Port Connectors must always be custom made, with any, or a combination, of these methods:

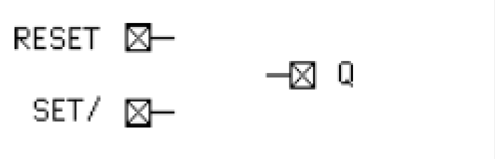
### Port/Pin Naming

The relationship between the port connector in the subcircuit and the pin on the parent device symbol is established by matching the pin name on the parent device with the Name of the port connector. For example, if we were to open the RSFF device used in the example earlier in this chapter using the DevEditor tool, we would see the following pins listed:



SET/
RESET
Q

For a complete port interface, a port connector must exist in the internal circuit named to match each one of these pins. In this case, the following port connectors would be required (ignoring all other internal circuitry):



The port interface is rechecked whenever any change is made. Thus, as soon as a port connector is added or removed, or its name changed, the port interface has been updated to reflect the new logical connections. However, to avoid excessive warning messages, error checking is performed only when an internal circuit is opened or closed. A warning box will be displayed if any error is found. This checking cannot be disabled.

**NOTE: The name of the port connector's pin and the name of the signal attached to the port connector are <u>not significant</u> in making the port association. Only the contents of the port connector's Name field is used. Note**

Using the auto-generate function that creates a subcircuit “shell” for you automatically when you create a new subcircuit. This is described in “Creating a Hierarchical Block - Top Down” on page 243.

Using the DevEditor tool to create a device symbol with the appropriate pins, either from scratch, or using an existing bus port connector as a guide. This method is described in “Creating Bus Ports” on page 265.

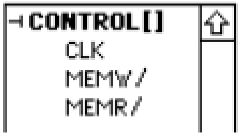
Using the Bus Pin Info command to modify and existing bus pin on a parent device and the corresponding port connector in the subcircuit. This is described in “Adding Internal Pins to a Bus Pin” on page 217.

**IMPORTANT: The Bus Port Connector <u>does not</u> export all the signals in the attached bus, <u>only</u> the ones for which is has explicit Bus Internal pins.**

### Bus Pin Example Using DevEditor

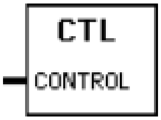
For example, the following simple device has a bus pin called CONTROL containing internal pins CLK, MEMW/, and MEMR/.

Pin List (in DevEditor)



CONTROL[]
CLK
MEMW/
MEMR/

Hierarchical Block Symbol



### Bus Pin Name Matching

Note the following rules for name matching in bus ports:

As with other Port Connectors, a Bus Port Connector must be given a Name exactly matching the pin name of the <u>bus pin</u> on the parent device.

The internal pins in the parent bus pin must exactly match the internal pins on the Bus Port Connectors bus pin.

The pin name of the bus pin itself on the Bus Port Connector is not significant.

As with normal ports, the names of the signals attached to the Bus Port Connector's pin are not significant.

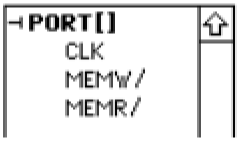
To create the corresponding Bus Port Connector using the DevEditor tool, follow these steps:

1) Select the DevEditor module from the Tools menu.

2) From the DevEdit menu select "Add Pins..."

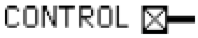
3) Enter the string "PORT[CLK MEMW/ MEMR/]". See “Adding Sequential Pin Names” on page 293 for Add Pins syntax for ranges of numbered signals, etc. You should now see a pin list like the following:

Pin List (in DevEditor)



PORT[]
CLK
MEMW/
MEMR/

Port Connector Symbol



4) For each bus internal pin set the *pin function* as appropriate, i.e. the <u>opposite</u> of the function of the parent pin. We will assume that the CLK signal is an input to the block and that MEMW/ and MEMR/ are outputs:

Open the pin info palette by double clicking on the first internal pin name from the DevEditor's pin list. In this example it is CLK.

Use the *pin function* pop-up menu located in the center of the dialog to select Output for the *pin function*.

Press the Enter key (not return) to step to the next pin in the list.

For the remaining pins MEMW/ and MEMR/ the default value of Input is correct since the parent pin is an output. Check that the remaining pins are correct.

5) Create a symbol for the connector, perhaps a simple rectangle.

6) Select the bus pin, PORT in the pin name list.

7) Place a bus pin from the DevEditor's tool palette.

8) Set the primitive type for the device.

Select "Subcircuit & Part Type..." from the DevEdit menu.

From the "Subcircuit & Part Type" dialog select "Primitive, Use Caution"

From the pop-up choose "PORT CONNECTOR".

Select "Done".

9) From the "File" menu choose "Save Part As..."

10) Choose or create a working library to save the part.

11) Close the DevEditor window.

12) Place your new Bus Port Connector in your internal circuit.

**NOTE: The *pin name* of the bus pin itself (in this case "PORT") is not important. The association between the Bus Port Connector and the parent bus pin is made by the name applied to the Bus Port Connector symbol itself. I.e. The same Bus Port Connector symbol can be used for any bus with the same internal signal names.**

**The comments under "Setting the Port Pin Type" on page 263 apply to each internal pin in a bus pin. The bus internal pins do not have to be the same. You can include any combination of names and functions in one bus pin.**

## Modifying an Existing Bus Port

An existing bus pin and its corresponding bus port connector can be modifed by either of these methods:

The parent symbol and the symbol for the bus port connector in the child circuit can be modified using the DevEditor tool using the usual symbol editing techniques. See "Editing an Existing Part on a Schematic" on page 289 and "Creating a Port Connector" on page 319.

The bus pin on the parent symbol can be modified using the Bus Pin Info command in the pin pop-up menu. This command allows you to add or delete internal pins in a bus pin and makes appropriate modifications to both the parent pin and the associated bus port connector. See "Adding Internal Pins to a Bus Pin" on page 217 for more information on this command.

## Making Power and Ground Connections Across Hierarchy Levels

Power and ground symbols (i.e. Signal Connector devices) do not make an immediate logical connection across hierarchy levels. For this reason, signal connectors should not be used to make active signal connections for interactive simulation purposes.

However, it is possible to merge power and ground nets across levels in netlist output using the Scripter tool’s $SIGSOURCE function.

## Using $SIGSOURCE and Device Attributes

Any number of special names can be specified as "signal sources" using the $SIGSOURCE keyword in a report form file. The Scripter searches the attributes attached to each device in a design for fields with these names. The pin number(s) specified in the value field of the attribute will be attached to a list for the signal of the same name.

For example, most of our library parts have a "Power" and "Ground" attribute.

In a circuit with two components U1, a 7404, and U2, a 74133:

> U1 has the attribute Power with a value of "14".
>
> U2 has the attribute Power with a value of "16".

In a report form the statement $SIGSOURCE(Power) will cause the Scripter to extract the pin number "14" from the device, say U1, and place it in the pin list for a signal called "Power".

```
PowerU1-14, U2-16
```

This ability is intended to allow power and ground nets to be creatted with out the necessity of having explicit power and ground pins and signals on every device. This should not be used for general signal connection.

## Using Signal Connectors

There are several pre-defined signal connectors in the Pseudo Devs library, Plus5V, Plus12V, Minus15V, etc...

You may create new ones using the DevEditor module. There are only two tricky parts to making a signal connector:

> Signal connectors can have only one pin, the name of the pin must be the name of the signal you wish to connect. For example to create a signal AGND the pin of the signal connector must be named AGND.
>
> You must set the primitive type of the new signal connector to "SIGNAL CONNECTOR" using the "SubCircuit & Part Type..."

dialog found in the DevEdit menu.

***Information on creating signal connectors with the DevEditor can be found in "Creating a Power (Signal) Connector" on page 318. Also see general information on using signal connectors in "Power and Ground Connections" on page 223.***

# Device Symbols

# 12

Device symbols are an important resource in your design creation process. Whether you primarily use the symbols provided with DesignWorks, or you create special libraries for your own use, the completeness and accuracy of this data has a major effect on your design flow. Library files generally outlast any one design and are used for many years across many projects. In addition, many DesignWorks features, for example, gate packaging, rely on specific steps being taken while creating a symbol. For these reasons, DesignWorks provides a variety of features for creating and editing the symbols themselves and for maintaining symbol library files.

This chapter covers these topics:

The creation and maintenance of symbol library files.

The creation and editing of individual device symbols using the DevEditor tool.

The attribute fields that are commonly used in symbols to implement gate packaging, auto-naming and other DesignWorks features.

How to use the standard libraries that come with DesignWorks.

Schematic operations that affect symbol definitions.

**IMPORTANT: Creating a device symbol can involve much more than just drawing the graphics that represent the device. Many DesignWorks features and many features in other ECAD packages that you may want to interface with, rely on correct text attributes being associated with each symbol. Before you start on a project that involves creating many symbol libraries be sure you read Chapter 6—Before Starting a Major Design on page 117. This will assist you in looking at your entire design process and the data that is required to make all parts of the process work correctly.**

# Working With Symbol Libraries

The symbols and related parameters for DesignWorks devices are stored in data files called symbol libraries. For each device type in a library the following data is stored:

> General information on the type, such as number of pins, number inputs, number of outputs, type name, default delay, default attributes, position, orientation and type of each pin, etc.
>
> A picture representing the symbol for this part type.
>
> A polygon outlining the symbol, used for highlighting and erasing the symbol.
>
> An optional internal circuit definition.

The following sections deal with the creation and maintenance of library files. Later parts of this chapter deal with editing the symbols themselves.

## Creating a New Library

A new, empty symbol library file is created using either of these methods:

1) Slide down the File menu to the Libraries sub-menu. In this sub-menu, select the New Lib command.
2) Hold the Control key pressed on the keyboard and click anywhere in the Parts palette. In the pop-up menu that appears, select the New Lib command.

In either case, a standard file save box will appear. Enter the desired name for the library and select a disk directory. The library created and

### Terminology Note

It is important to distinguish between the *definition* of a symbol that is stored in a library, and an *instance* or usage of it in a schematic. In this manual we use several different terms to refer to the definition of a device symbol in a library. If we use the term symbol, part, part type, or just type, we are referring to the definition of a device symbol in a library. The term symbol is used when we are primarily interested in the graphical representation of a device, but a symbol is always stored with related pin definition information and text attributes.

opened automatically so it appears in the Parts palette. If you wish to have the library opened automatically at startup when you enter DesignWorks in the future, see “Specifying Libraries to Open at Startup” on page 393.

***If you are creating a library as part of a design process that will be used over and over, you may wish to make the library part of a design kit. Refer to Chapter 13—Design Kits and Sheet Templates on page 335 for the basics of creating design kits.***

## Designs and Libraries

Whenever select a device symbol from a library and use it in a design, all information needed to display and edit that device is retained with the design. (Of course, only one copy is kept, regardless of how many times you use the same symbol in one design.) No further access to the library itself is required. This is done to ensure that a design file is always a complete entity and that future changes to a library will not inadvertently render an old design incorrect.

However, there are frequently times when you would like to update a symbol in a library and then copy the changes into one or several designs. Conversely, you may have edited a symbol or its subcircuit on a single design and wish to save the changes back to a library for use elsewhere.

For these reasons, DesignWorks does retain some information with each symbol about its “home” library and provides a number of features to allow transfer and updating of symbols between designs and libraries. Each time a new symbol is used in a design (i.e. one that hasn’t been used in this design before) the library name, file path and “last modified” date are stored in the attribute fields LibName, LibPath and LibDate, respectively. These are used by the Update from Lib command to locate the original library and determine if it has changed since we used it.

In addition, whenever the definiton of a device symbol is modified, a new “checksum” value is calculated. This is also used as a check to verify the equivalence of two symbols whenever one is loaded from a library. This checksum cannot be set by the user, but it can be dumped in report using the $CHECKSUM script command, described in the DesignWorks Script Language Reference (separate manual on disk). This can be used to tell if inadvertant modifications have been made to a symbol, but not to another that appears similar.

Two utility tools are provided with DesignWorks to allow the mass transfer of all the symbols in a design to a library, or to place one each of all the symbols in a library in a design, for testing or modification. These tools are described in “Copying Symbols

## Manually Opening a Library

To open any library file on your disk, either:

Slide down the File menu to the Libraries sub-menu. In this sub-menu, select the New Lib command,

or,

Hold the Control key pressed and click anywhere in the Parts palette. In the pop-up menu that appears, select the New Lib command.

In either case, a standard file open box will appear. Locate the desired file in the usual way. The library will be opened and appear in the Parts palette. A small amount of memory is occupied by each open library file.

## Automatically Opening Libraries at Startup

Libraries can be opened automatically when the program starts by any ofthese methods:

Placing the library (or an alias for the library) in the Libraries folder of any installed design kit. If the library in question forms a logical part of the design flow for a particular application, then it is best to group it with other items like scripts and templates that form part of the design process, and create a design kit.

Placing a command in the DesignWorks Setup file to specifically open the library using the LIBRARY or LIBRARYFOLDER setup file keywords. See “Specifying Libraries to Open at Startup” on page 393 for more information on this.

***See Chapter 13—Design Kits and Sheet Templates on page 335 for more information on creating and installing design kits and using the setup file commands.***

## Manually Closing a Library

To close any open library file, either:

Slide down the File menu to the Libraries sub-menu. In this sub-menu, select the Close Lib command,

or,

Hold the Control key pressed on the keyboard and click anywhere in the Parts palette. In the pop-up menu that appears, select the Close Lib command.

In either case, a list of the open library files will appear. You can use the SHIFT and cmd keys to select multiple files to be closed in one operation and then press the Close Lib button. Alternatively, you can simply double-click on the name of a single library.

**NOTE: Any information required for symbols used in any open designs will be automatically retained in memory. Once you have used a symbol in a design, all information required has been copied into the design's data. No further access to the library itself is required.**

## Copying Symbols from One Library to Another

To copy one or more symbols from one library to another, follow these steps:

Make sure the source and destination libraries are open in the Parts palette. If not, follow the steps under "Manually Opening a Library" on page 274.

Select the Lib Maint command, either in the Libraries sub-menu of the File menu, or by holding the Control key pressed while clicking in the Parts palette.

Select the source library in the pop-up library selection menu above the "Souce Lib" list.

Select the destination library in the pop-up library selection menu above the "Dest Lib" list.

Select the symbols to be copied in the source list. You can select multiple items using the Shift and Command keys.

Click the Copy button.

The copy operation will now proceed, with status messages appearing in the Messages area at the bottom of the box.

## Copying Symbols from a Design to a Library

See "Saving a Symbol Definition from a Schematic to a Library" on page 282.

## Deleting Symbols from a Library

One or more symbols may be permanently deleted from a library by following these steps:

Make sure the target library is open in the Parts palette. If not, follow the steps in “Manually Opening a Library” on page 274.

Select the Lib Maint command, either in the Libraries sub-menu of the File menu, or by holding the Control key pressed while clicking in the Parts palette.

Select the target library in the pop-up library selection menu above the “Souce Lib” list.

Select the symbols to be deleted in the source list. You can select multiple items using the SHIFT and Command keys.

Click the Delete button. You will be prompted to confirm the operation before the items are permanently deleted.

**WARNING: The Delete operation cannot be undone!**

## Duplicating a Symbol Within a Library

You can duplicate a symbol within a single library by following these steps:

Make sure the target library is open in the Parts palette. If not, follow the steps in “Manually Opening a Library” on page 274.

Select the Lib Maint command, either in the Libraries sub-menu of the File menu, or by holding the Control key pressed while clicking in the Parts palette.

Select the target library in the pop-up library selection menu above the “Souce Lib” list.

Select the symbols to be deleted in the source list. You can select multiple items using the SHIFT and Command keys.

Click the Duplicate button. You will be prompted for each selected item to enter a new name. Names must be unique within a library.

## Renaming a Symbol in a Library

You can rename a single symbol in a library by following these steps:

Make sure the target library is open in the Parts palette. If not, follow the steps under "Manually Opening a Library" on page 274.

Select the Lib Maint command, either in the Libraries sub-menu of the File menu, or by holding the Control key pressed while clicking in the Parts palette.

Select the target library in the pop-up library selection menu above the "Souce Lib" list.

Select the symbols to be renamed in the source list. You can select multiple items using the SHIFT and Command keys.

Click the Rename button. You will be prompted for each selected item to enter a new name. Names must be unique within a library.

## Reordering Symbols Within a Library

Several options are available in the Library Maintenance box for reordering symbols within a library. These options do not affect the data associated with any symbol, they merely change the order in which they appear in the Parts palette.

First, to display this box, select the Lib Maint command, either in the Libraries sub-menu of the File menu, or by holding the Control key pressed while clicking in the Parts palette. Two kinds of reordering operations are available:

The Promote and Demote buttons cause the selected items in the Source Lib to be moved up or down the list, respectively.

The Sort button sorts the entire list alphabetically, with the numeric part of any name treated as an integer. The adjacent arrow buttons determine the direction of the sort.

## Compacting a Library

When parts are deleted from a library, the free space in the file is not automatically recovered. In most cases this is not a significant overhead. However, if a large percentage of the parts in a library have been deleted then you may wish to compact the file. To do this:

Create a new, empty library which will become the target for the Compact operation.

Select the Lib Maint command.

Select the library to be compacted as the Source Lib.

Select the new, empty library as the Dest Lib.

Click on the Compact button.

**IMPORTANT: Verify that the new destination library is correct before discarding the old copy.**

# Operations on Symbols in a Schematic

## Making a Single Device Into a Unique Type

This command makes the selected device into a unique type. In other words, even if other devices in the same design were originally drawn from the same library, this one will now be considered to be unique. This is primarily intended for hierarchical designs. This allows the internal circuit for the selected device to be modified or deleted without affecting other similar devices.

As an example, you could create a standard template for a Programmable Logic Device in a library, complete with internal simulation circuit. Several of these devices could be placed in the design, even though they will eventually have different internal programming. The Make Unique Type command would then be used to isolate them so that their internal circuits could be separately updated.

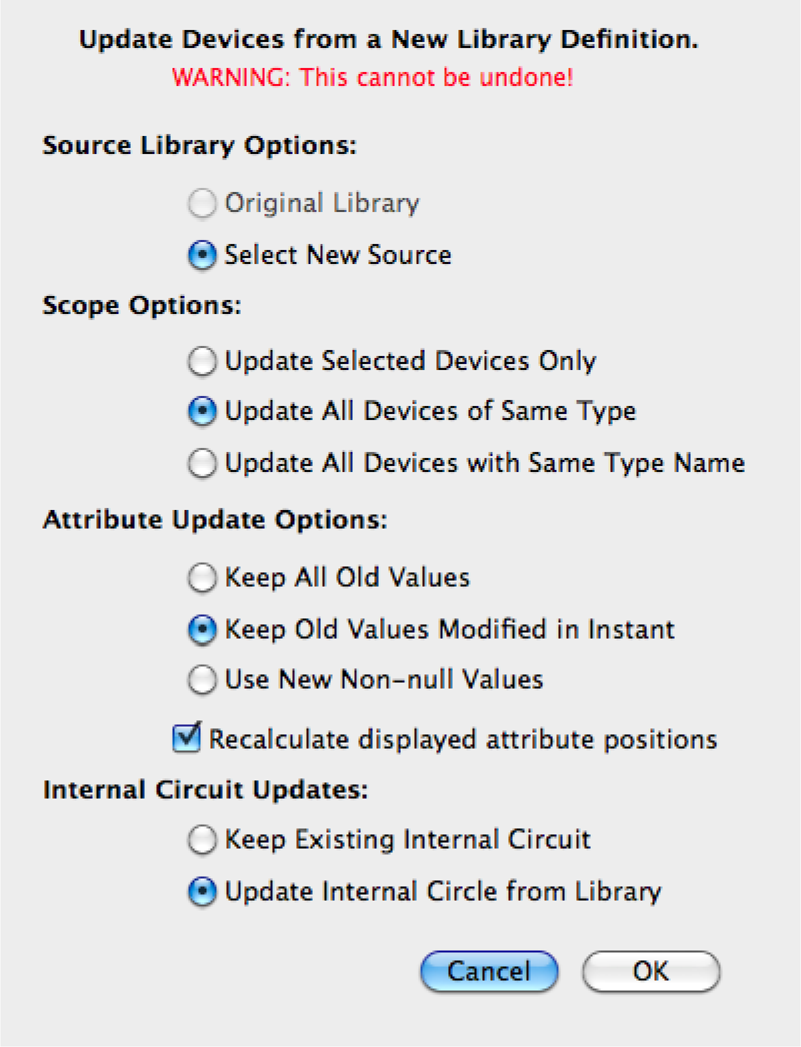
***See more information on this in the sections "Associating a Subcircuit with a Device Symbol" on page 252.***

## Updating a Symbol from a Library

The Update from Lib command updates the definition of the selected device from its original source library. There are a variety of options available to determine how attributes and sub-circuits are updated and how a source symbol definition is selected.

**WARNING: Depending upon the options selected, this command may replace the entire internal circuit of the selected device. THIS CANNOT BE UNDONE!**

Selecting the Update from Lib command in the Part Type sub-menu of the Options menu causes the following box to be displayed:



**Update Devices from a New Library Definition.**

WARNING: This cannot be undone!

**Source Library Options:**

○ Original Library

● Select New Source

**Scope Options:**

○ Update Selected Devices Only

● Update All Devices of Same Type

○ Update All Devices with Same Type Name

**Attribute Update Options:**

○ Keep All Old Values

● Keep Old Values Modified in Instant

○ Use New Non-null Values

☑ Recalculate displayed attribute positions

**Internal Circuit Updates:**

○ Keep Existing Internal Circuit

● Update Internal Circle from Library

The first option pop-up menu allows you to select how the source library will be located:

| | |
|---|---|
| **Original Library** | This selection will locate the original library that the device was derived from using information stored in its attribute fields. As long as the library can be located and a part with the same name can be located in the library, you will not be prompted to locate a source. |
| **Select New Source** | This selection will cause a sequence of two prompt boxes to be displayed requesting a library and then a part within the library. This new symbol will be used to update the selected device. |

Next, you need to select which devices on the schematic you wish to update. The choices are:

| | |
|---|---|
| **Update This Device Only** | Only the selected device will be updated. NOTE: This in effect redefines the device to be different from others that were originally derived from the same library part. |
| **Update All Devices of Same Type** | This will update all devices in the design that were originally derived from the same part type. Note that this is determined by the internal logical linkage between the device instances and the part definition, <u>not</u> by the type name. If you have multiple type definitions with the same name in your design, this option will only update the devices linked to the same type as the selected one. |
| **Update All Devices with Same Type Name** | This option will update all devices derived from a type with the same name as the selected one, even if the definitions have become separated by some previous operation. This is specifically intended to remedy the case where a design inadvertently contains multiple definitions of similar devices. |

The Attribute Update Options pop-up menu allows you to choose how attribute values are updated in the event that the new symbol definition has different values than the existing device instance:

| | |
|---|---|
| **Keep All Old Values** | This selection tells DesignWorks to maintain all attribute values exactly as they are in the current instance, even if the new symbol definition has different values. |

| | |
|---|---|
| **Keep Old Values Modified in Instance** | This selection indicates that you wish to keep any attribute values in the device instance that were changed from the default value in the original definition. Any values that still have the default specified in the old symbol will be updated to the new default value in the new symbol. |
| **Use New Non-null Values** | This item indicates that you wish to use the value specified in the new definition, if there is one, but keep any old values that are not overridden by new ones. |
| **Use All New Values** | In this case, the updated instance will have exactly the values in the new definition, even if they are empty. All attribute data present in the old definition will have been replaced by new values. |

**NOTE: The Name and InstName fields are not modified by the Update operation, regardless of attribute update setting.**

In addition to the attribute value options, the "Recalculate displayed attribute positions" checkbox allows you to determine whether displayed attribute values will be left at their old positions or recalculated to fit with the symbol. If the symbol has not changed in shape appreciably and you have manually repositioned the displayed values, you may wish to turn off this option.

Finally, if either the existing instance or the definition in the library, or both, have sub-circuits, then you need to choose which circuit you wish to keep.

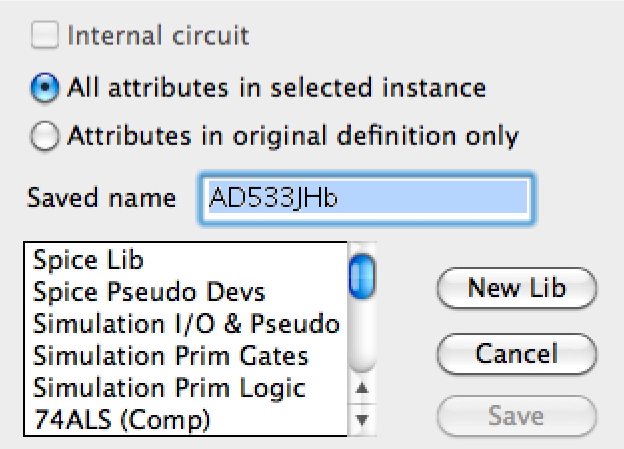
| | |
|---|---|
| **Update Internal Circuit from Library** | If the selected device has an internal circuit it is removed. If the new library part definition has an internal circuit as new one is created. |
| **Keep Existing Internal Circuit** | Any existing internal circuit is kept and only the symbol is updated. Note that if you make this selection, the part definition in the design will be marked as having been changed since it is not the same as the library definition. |

**IMPORTANT: Updating Devices in a Hierarchical Design - If your design is in Physical hierarchy mode and you are editing a sub-circuit whose parent device is instantiated multiple times, devices that are in other instances of the same sub-circuit will not be updated immediately, even if you select "Update All Devices of Same Type". Only the open copy of**

**the circuit will be updated immediately on execution of the command. The other copies of the circuit will be updated when you close the sub-circuit and select the Update option.**

## Saving a Symbol Definition from a Schematic to a Library

The Save to Lib command saves a type definition for the selected device to a library. The following box will be displayed:



☐ Internal circuit

● All attributes in selected instance

○ Attributes in original definition only

Saved name AD533JHb

- Spice Lib
- Spice Pseudo Devs
- Simulation I/O & Pseudo
- Simulation Prim Gates
- Simulation Prim Logic
- 74ALS (Comp)

This table summarizes the options available.

| | |
|---|---|
| **Internal Circuit** | If this box is checked, the internal circuit attached to the selected device will be saved with the part definition. |
| **All attributes in selected instance** | If this option is selected, all the attribute values associated with the selected device will be made part of the saved library part. |
| **Attributes in original definition only** | If this option is selected, only attributes that were originally defined for the library part will be saved. |
| **Saved Name** | The part name under which the new library entry will be saved. |
| **New Lib** | This button will display a standard file save box allowing you to create a new, empty library. |

## Saving All the Symbols in a Design to a

## Library

To make a library containing all the symol definitions used in a circuit:

> Ensure that the circuit whose symbols you want to save are in the frontmost window.
>
> Ensure that the library you wish to write the symbols to is selected in the Parts palette. If it is not, either select it from the library selection menu in the Parts palette, or use the New Lib command to create one.
>
> Select the CctToLib item in the Tools menu.
>
> Click on the OK button.

The CctToLib tool will now proceed to write all the unique symbol definitions in the circuit to the selected library. Here are some points to note on the operation of this tool:

> In a hierarchical design, only symbols in the current circuit level are written.
>
> Pseudo-devices (i.e. things like power and ground symbols and page connectors) are included in the items written out. Depending on your requirements, you may wish to delete these from the final library.
>
> The symbol written out will include all attribute values defined in the first instance of the symbol that is encountered, except for Name and InstName. I.e. The symbol that is written out is not necessarily identical to the definition used in creating the circuit in the first place.

**TIP: One of the purposes of CctLib is to allow you to use the attribute editing tools like the Browser to work on large numbers of symbols at once. You can use LibToCct to create a design containing one of each symbol in a library, then editing the resulting devices and write the modified versions back out. This can be very convenient, for example, if you need to add the same attribute value to all the symbols in a whole library. See "Creating a Design With One of Each Symbol in a Library" on page 284 and "Using the Browser Tool" on page 159.**

## Editing a Device Symbol in a Schematic

See "Editing an Existing Part on a Schematic" on page 289.

## Creating a Design With One of Each Symbol in a Library

The are a number of cases where it is useful to create a design that contains one of each symbol in a library, such as:

To test a library to ensure that all symbols appear, print and netlist correctly.

To perform mass modifications on part attributes using the schematic tools such as the Browser. See also "Saving All the Symbols in a Design to a Library" on page 282.

To perform this operation, follow these steps:

Create a new, empty design using any desired template.

Open or select the desired library so that it is the current one displayed in the Parts palette.

Select the LibToCct item in the Tools menu.

Click the OK button.

The LibToCct tool will proceed to place symbols on the schematic in columns based on the sizes of the symbols. If it runs out of room on the page, it automatically creates a new circuit page. The naming of the devices and display of attributes are determined by the template's settings in the same way as if you had placed the parts manually.

**WARNING: This process can require a large amount of memory. Because every device in the resulting design has a unique symbol definition, a large amount of data is loaded into memory while the design is open. As a rule of thumb, allow 10K of free memory (as displayed in the schematic tool palette free memory display) for each symbol in the library.**

## Editing Device Symbols

Symbols are created and edited using the Device Editor tool. In addition to drawing symbols, the DevEditor can also be used for general graphics (e.g. title blocks or simple mechanical drawings) for use on DesignWorks schematics. It provides a complete, object-oriented drawing environment with standard drawing tools, as well as specific functions tailored for symbol creation.

## Creating a New Part from Scratch

To create a new device symbol with no initial attribute settings or graphics:

> If another DevEditor window is already the frontmost window, then select the New Part command in the File menu.
>
> If some other window is frontmost, or no windows are open, select the DevEditor command in the Tools menu. If there was already an open DevEditor window, this command will bring it to the front and you can then select the New Part command in the File menu. If no other DevEditor window was open, an empty DevEditor window will appear.

**NOTE: In many cases, you may wish to start with an existing symbol that has settings similar to the one you require, rather than creating a complete new one.**

**IMPORTANT: The procedure given here allows you to produce a symbol with only the simplest graphical and netlisting requirements. Details of creating the symbol and the various settings that may be required for specific applications are covered in later sections of this manual, and in other chapters on specific DesignWorks functions. You may wish to refer to any of these sections:**

> A simple procedure for creating simple schematic symbols and hierarchy blocks is given in the tuturial section entitled “Device Symbol Editing and Hierarchical Design” on page 42.

### Using the Clipboard in DevEditor

The standard Edit menu commands Cut, Copy and Paste can be used to move objects inside and between the DevEditor window, DesignWorks circuit windows, and other applications. Some types of graphic objects, notably bitmaps, created by other programs are not supported by the current version of DevEditor and will not appear if pasted into the DevEditor workspace area.

For details on the drawing tools available for creating symbols, see “Drawing the Graphics” on page 312.

For information on the general entry of part and pin attributes, see “Setting Part and Pin Attributes” on page 295.

For pin settings and how they may affect simulation or netlists, see “Displaying the Pin Info Palette” on page 305.

For information on creating symbols for hierarchical blocks, see “Creating a Block Symbol” on page 243.

For details on adding gate packaging information to a symbol, see “Gate Packaging” on page 320.

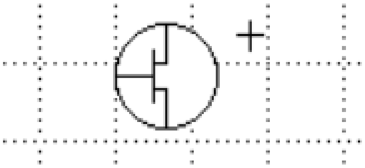
For assistance in determining what information should be in a symbol to ensure correct netlist generation and interfacing, see Chapter 6—Before Starting a Major Design on page 117.

For information on creating pseudo-device symbols, see “Creating a Power (Signal) Connector” on page 318 and the sections that follow it.

### Drawing the Graphics

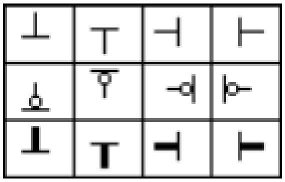
Draw the graphical shape which represents the part. Do this using the line, rectangle, rounded-rectangle, oval/circle, arc, and polygon tools.

**NOTE: Device pins must be added using the pin tools. Do not draw any of the part's pins with the basic graphic tools.**



### Adding Pins to the Symbol

Place pins on the part use the 12 pin placement tools:

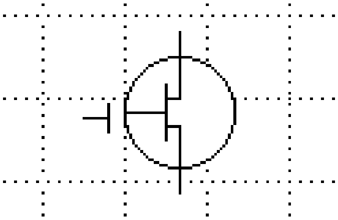


This palette provides three types of pins:

The first row allows placement of normal pins (i.e. not bus or inverted).

The second row provides pins with inversion bubbles. Note that the only difference between these pins and the normal ones is graphical. Using a pin with an inversion bubble does not affect its operation in the DesignWorks Simulator, nor does it affect its pin type or any other attributes.

The last row of pin tools are used to place bus pins.

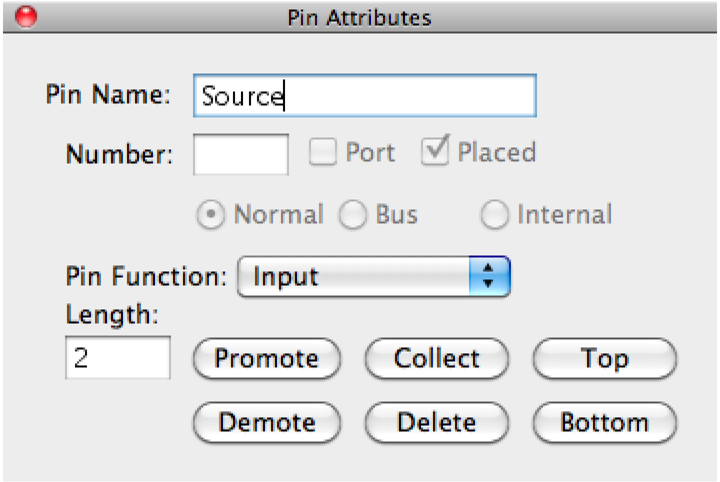


Note that each time you place a pin a new default pin name is added to the pin list. See “Entering Pin Information” on page 303 for more information.

**IMPORTANT: Some netlist formats, notably SPICE, require that pins appear in a certain order in the output. Unless you specify otherwise using attributes, this order will be determined by the order in which the pins are placed on the symbol. See Chapter 6—Before Starting a Major Design on page 117 for more information.**

The pin name and other pin information can be changed using the pin info palette (See “Entering Pin Information” on page 303). For this example we want to name the pins "Source", "Gate", and "Drain". In addition we want to give them the following pin number "S", "G", and "D".

To change the information about each pin we first bring up the Pin Info palette by double clicking on a pin name in the pin list. Double click on the first pin. When placing the pins we placed the top pin, then the bottom pin and finally the pin on the left. Therefore, the names in the pin list are in the same ord80er.



Pin Name: Source

Number: ☐ Port ☑ Placed

● Normal ○ Bus ○ Internal

Pin Function: Input

Length: 2

Enter the name "Source" and the number "S" for the first pin we placed. You can use the tab key to move between the name and the number field. When you have entered both the name and number - Press the Return key. Notice that the name was updated in the pin list and that the pin number "S" has been displayed with the graphic pin.

To change the next pin you can; click on the graphic pin; click on the pin name in the pin list; or if the Pin Info palette still has the text focus press the Enter key. Press the Enter Key. Pressing the enter key also causes the pin information to be copied to the pin, just like pressing the return key.

Notice that the pin whose information is displayed in the Pin Info palette is also selected in the pin list and if it has been placed then its graphic image is also selected. This allows easy verification that the correct pin is being renamed.

You should now see PIN2 selected in the pin list, the bottom graphic pin selected, and the name PIN2 in the Pin Info palette. Do as describe above for each of the remaining pins.

When you are finished with the Pin Info palette it may be simply closed by clicking on its go-away box or since it is a floating window it can be just moved to the side.

The actual names given to the pins in this simple form of part are not critical. It is a good idea though to use meaningful names for these reasons:

> These names will be seen in the Schematic tool's pin info dialog.
>
> They can be extracted in netlist output.
>
> They are used when binding pins to ports in subcircuits.
>
> In the case of bus internal pins, they are used when connecting bus pins on the part to busses.

## Editing an Existing Part in a Library

The following methods can be used to edit an existing part in a library:

> A part selected in the part palette may be edited by pressing the mouse button with the Control key down and then selecting "Edit Part" from the pop-up menu. The mouse must be over the parts list for the pop-up menu to appear. Another place to find this menu is under the Libraries sub-menu in the File menu.
>
> Open the Device Editor using the DevEditor command in the Tools menu, select a part in the Parts palette, then select the Open Part command in the File menu.

In response to either of these operations, a copy of the symbol definition is loaded into the DevEditor window. No changes to the source library will be made until you save the open symbol back to its original library.

**NOTE: Editing a symbol in a library does not automatically update designs that have used that symbol. For more information, see “Designs and Libraries” on page 273.**

## Editing an Existing Part on a Schematic

You can edit an existing symbol right on the schematic by holding the Control key pressed while clicking and holding on the device in question. In the device pop-up menu, select the Edit Symbol command. This actually performs the following steps:

> Saves the symbol definition to a temporary library, equivalent to doing a Save to Lib on the device.

Stores information about the location of the selected device in a temporary design attribute.

Opens a DevEditor window on the saved symbol.

If you close the DevEditor window without saving the symbol back to its original library, no further action is taken. However, if you modify and Save to the part, you will be prompted to update the device on the schematic from the library. This prompt is actually a simplified version of the Update from Lib command, described in "Updating a Symbol from a Library" on page 278. If you choose to update the device, all other devices of the same type will also be updated. If the device has an internal circuit, this will be retained. If any attributes had been entered for the selected device instance these changes will be retained.

The "More Options" button on the Update box allows you to switch to the full Update from Lib box to get access to more update options.

**TIP: If you only want to modify the pin connections in a bus pin and don't need to modify the graphics of the symbol or its attributes, you can use the Bus Pin Info command described in "Using Bus Pins" on page 214.**

## Closing the DevEditor Window

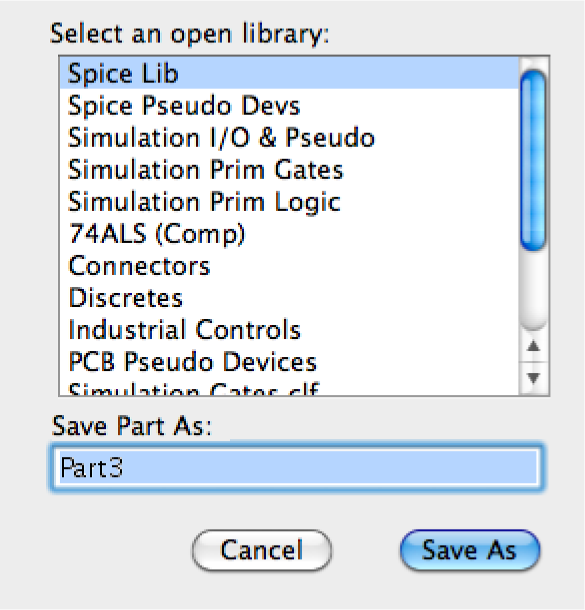
An open DevEditor windows can be closed by either clicking in the close box at the upper left corner or selecting the Close Part command in the File menu. If any changes have been made to the open part, you will be prompted to save or discard the changes.

## Saving an Edited Part Back to its Original Library

The Save Part command saves the contents of the current DevEditor window back to the library it was read from. If the open part was not read from a library (i.e. it was just created), this item will be disabled.

## Saving the Part Under a New Name

To save an open symbol under a new name, or to a library other than the one from which it was read, select the Save Part As command in the File menu.



Save Part As: Part3

When you select this menu item, the "Save Part As" dialog will appear. It requires that a library name be selected from the list and that a name for the part be entered in the lower box. In this example the Connectors library has been selected and the part's name has been left to default to "Part1". A name entered will become the name of the part in the library, and the title of the DevEditor window will be updated to correspond to the new part name.

**NOTE: Only open libraries appear in the list. See "Manually Opening a Library" on page 274 for information on opening libraries.**

## Zooming the DevEditor Window

The Reduce/Enlarge/Normal Size commands in the Drawing menu allow you to adjust the scale at which an object is viewed. The default setting for the DevEditor is to display objects at the same size as they will appear in the schematic at Normal Size. There are five display levels: 50, 100, 200, 400 and 800%.

## Warnings About Editing an Existing Symbol

### Changing Graphics

Editing changes can be freely made to all of the part's graphical components other then pins. It is important to remember that graphic pins are associated with logical pins in the pin list and that even though graphic pins look the same they may not be interchangeable because of their associated name and attributes. To ensure that pins are placed correctly it is important to select the pin in the pin list and verify that the expected graphic pin is selected.

Consider the following example. A flip flop has pins named "D", "CLK", and "Q". The pins "D" and "CLK" are both left facing graphic pins and "Q" faces to the right. Simulation and netlisting problems will occur if the graphic pins "D" and "CLK" were to have their graphical locations swapped.

Swapping the pins' positions would be confusing because the "D" label would be located next to the graphical pin associated with the pin name "CLK".. The same would be true for the "CLK" pin.

### Changing Pin Information

If the part has a subcircuit then it is important to remember that the pin types are derived from the ports in the subcircuit. Trying to change a pin's type without also changing the underlying subcircuit is not allowed.

For parts based on simulation primitive types, the functions of the pins must agree with the selected model. The program does not check to ensure that the pins are set correctly for the model selected.

### Changing Part Type

Changing a part's primitive type can have dramatic effects. Some of the things to look out for are:

> Changing an internal subcircuit to any other type. An internal circuit is kept with the part in the library, it is important to remember that changing the part's type will cause the circuit to be deleted.
>
> Changing to an internal or external subcircuit. When a circuit is associated with the part its ports are read and used to create pins with the same name. These new pins carry the type of the port and will replace like named pins that are already defined. This will result in a pin losing the type it was initially defined with.

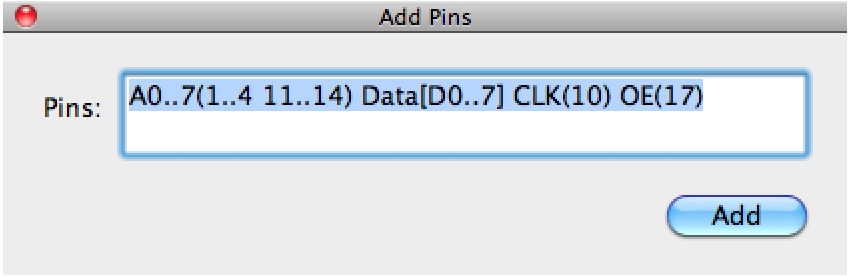
## Adding Sequential Pin Names

The basic procedure for adding pins to a symbol is described in “Adding Pins to the Symbol” on page 286. However, the basic procedure requires that the pin name and number be entered manually for each pin, which can be rather tedious for large numbers of sequentially numbered pins. To solve this problem, the Add Pins command allows you to enter a sequence of names in a format similar to that used for bus breakout creation and for the Auto Create Symbol command. These pins are added to the pin list, but not placed on the symbol. After the pins have been added to the pin list, you can place the pins sequentially on the graphic with no further typing required.

**TIP: You can use the Auto Create Symbol command to specify the pin names and auto-generate a rectangular symbol in one operation.**

The Add Pins command causes a floating palette to be displayed. Pin names you wish to add to the pin list, and optionally their pin number, may be entered into the palette. The pins are created and merged with the contents of the pin list when a carriage return or enter key is pressed or when the Add button is clicked. The Add Pins palette does not create graphic pins, only pins for the pin list.

The created pins are merged with the names in the pin list. If a like named pin already exists in the list then it may be reordered to appear in the same order as in the Add Pins palette. If a pin being added has a pin number defined then this pin number will replace the pin number in the like named pin.



Pins: A0..7(1..4 11..14) Data[D0..7] CLK(10) OE(17)

### Syntax for Pin Names in Add Pins

Here are the rules that outline how a list of pin names is entered:

Pin names may be up to 16 characters long.

Normal pin names (that is, not bus pins or bus internal pins) may be specified as individual names, e.g.: A B C D, or as sequences, e.g.: A0..3. After each pin name or pin name sequence an optional pin number specification is allowed.

A pin number specification defines the pin number(s) associated with the proceeding pin name or pin name sequence. The specification starts with an **(** and ends with an **)**. For a single pin name the pin number specification should contain a single number. For a pin name sequence multiple numbers and sequences may appear between the brackets.

In the picture above CLK and OE both define normal pin names which have pin numbers 10 and 17, respectfully. The sequence A0..7 defines the pin names A0 through A7. These pin names were also given pin numbers. The pin numbers 1 2 3 4 11 12 13 and 14 were assigned.

It is also possible to skip a pin name when assigning pin numbers to a pin name sequence. Consider the previous example, if we didn't want to assign the number 11 but wanted all other pin names to get the same number, we would do the following: A0..7(1..4,,12..14) instead of A0..7(1..4 11..14). Two commas in a row causes a pin to be skipped when assigning the pin numbers. Three commas cause two pins to be skipped. Four commas, etc.

Bus pin names are denoted by a name followed by **[...]**. Bus pins do not need to have internal pin names defined and may not have pin numbers.

Bus internal pin names must appear between a **[** and a **]**. They have the same format as normal pin names and may have pin numbers. Internal pin names may be added without adding a bus name pin by not placing a bus name in front of the **[]**. For example, [A B C] adds the internal pin names "A", "B", and "C" to the pin list.

## Deleting Pins

In a DevEditor window, pins exist both in the graphic representation of the symbol and in the pin list. They have to be deleted from both places before they are completely removed from the part definition.

To remove a graphical pin from the symbol, simply select it by clicking on it and delete it with the delete key in the usual way. Note that this does not remove the corresponding entry from the pin list. If you wish to replace the pin, for example to use a different orientation, you can simply select the item in the list and place the appropriate pin graphic from the palette.

To remove the selected pins in the pin list and their associated graphic pins:

Double-click on any pin in the pin list to display the pin info palette, if it is not already.

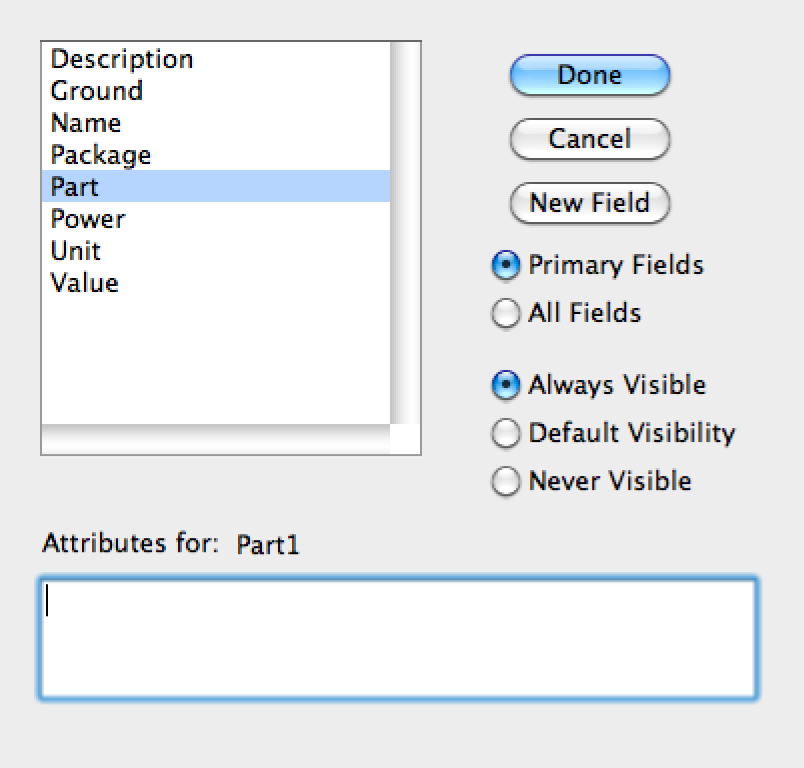
Select the pin or pins in the pin list that you wish to delete. You can select more pins by holding the Shift or cmd keys while clicking in the list.

Click on the Delete button in the pin info palette. It will display a confirmation box before proceeding to delete the pin entries.

## Setting Part and Pin Attributes

To edit the attributes associated with a part select the Part Attributes command in the DevEditor menu. To edit the attributes associated with a pin, select a pin in the pin list and then select the Pin Attributes command from the DevEditor menu. The only visible difference between the part attribute dialog and the pin attribute dialog is the addition of "Next" and

"Previous" buttons at the bottom of the window.\



● Primary Fields
○ All Fields

● Always Visible
○ Default Visibility
○ Never Visible

Attributes for: Part1

Attributes defined while in the DevEditor will be associated with the part in the library. These will become the default attributes for the part when it is placed in a circuit.

**IMPORTANT: Only attribute fields that have a value are stored with the symbol. When you save the symbol to a library, any fields with null values are stripped off to save storage. This has the side effect that you cannot create a "place holder" field definition in a symbol for future use without putting a value in it.**

Some visibility options appear in this version of the attributes box that are different from what you will see on a device placed in a schematic:

| | |
|---|---|
| **Always Visible** | This setting indicates that the select field should always be made visible on the schematic when the device is placed, regardless of the visibility setting for this field in the target design. This setting does not prevent the field from being removed after the device is placed, it just sets the initial state. |
| **Default Visibility** | This setting indicates that we wish to make the field visible only if it is defined as Visible by Default in the design in which it is placed. |
| **Never Visible** | This indicates that the field value should not be displayed when the device is placed, regardless of the design's Visible by Default setting for this field. This does not prevent the value from being displayed later, it just sets the initial state. |

**TIP: You can set the default visibility independently for each attribute field defined in a design. This is done by selecting the Define Attribute Fields command in the Options menu**

### Attribute Field Definitions in Symbols and Designs

In DesignWorks, symbol definitions are completely independent of any particular design or library file. For this reason, symbols carry around their own definitions of the attribute fields that are associated with them. By default, when you create a new symbol, you see the attribute definitions for the design that is currently open, or a default set if no design is open. However, the New Field button in the Part Attributes box allows you to add new field definitions that affect only this one part definition. This operation does not automatically add the same field to the current design.

When you place a symbol in a design for the first time, the program compares the attribute fields defined in the symbol to those defined in the design's symbol table. If there are any new fields in the symbol, you will be asked if you wish to add them to the design's table. Your choices in this case are to either:

> Click the OK button and allow the program to automatically add the field definitions to the design's attribute table. The program will attempt to make reasonable settings for the maximum length, default visibility and other settings, based on the current value, or,
>
> You can click Cancel to stop the device placement and add the fields manually to the design's table using the Define Attribute Fields command. In general, this is preferred, since it gives you more control over the option settings for the field.

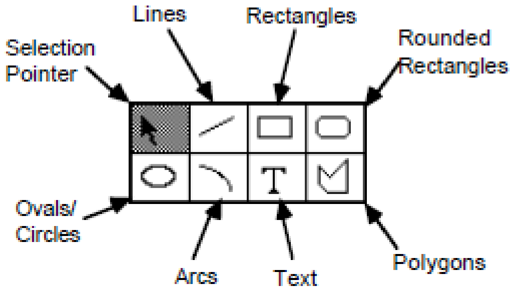
**and setting the "Visible by default" option as desired for each field.**

For more information on how to create and modify attributes see “Features Requiring Symbol Attributes” on page 320.

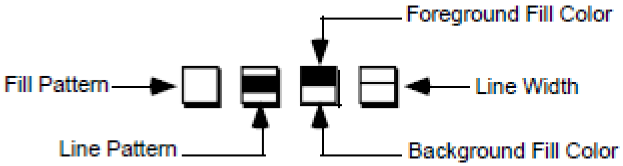
# Editing Symbol Graphics

## Using the DevEditor Drawing Tools

The 8 items in the top half of the tool palette represent standard Macintosh drawing tools.



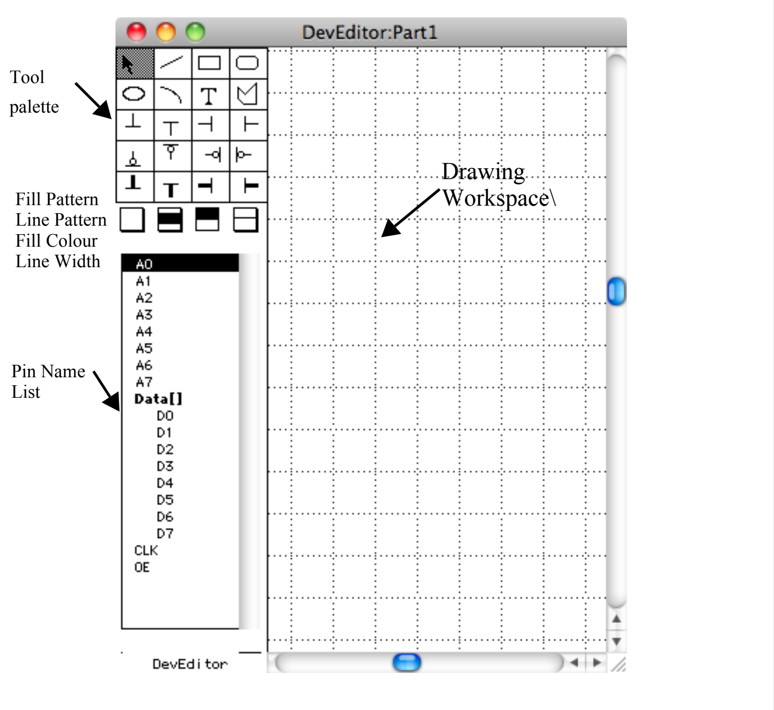
Lower on the DevEditor window are the graphic palettes used to select colours, line widths and fill patterns:



These palettes change how the selected graphical objects are drawn. If there are selected objects when a palette is changed then the selected objects will be given the characteristic selected. If there are no graphical objects selected when a change is made to a palette, the default setting for the palette is changed. Graphical objects created in the future will use the new default setting.

# The DevEditor Window

The DevEditor tool has its own complete drawing envrionment and tool set in its window. The table below describes all the tools available for editing device symbols.



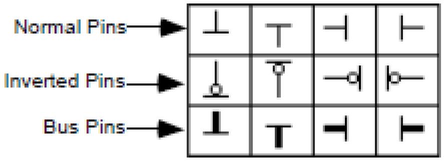
| | |
|---|---|
| **Window Title** | "DevEditor:" followed by the name of the part being edited. |
| **Tool Palette** | Used to select a graphical editing or pin placement tool. The current tool is displayed selected. |
| **Fill Pattern** | Pop-up palette. Graphic objects are filled with this pattern. |
| **Line Pattern** | Pop-up palette. Graphic objects have their edges drawn with this pattern. |
| **Fill Color** | Split pop-up palette. The top half of the pop-up is used to set the foreground color of a graphic object, the bottom half of the palette sets the background color for an object. |
| **Line Width** | Pop-up palette. Used to select the width of lines. |
| **Pin List** | Scrolling list of names for all of the pins which exists. A pin exists as long as it is in the pin list. It need not have a graphic representation to still be valid. |
| **Drawing Workspace** | This area is used to construct the graphical appearance of the part. |

The following table summarizes the usage of these tools:

| | | |
|---|---|---|
| **Fill Pattern** | ☐ | The fill pattern palette determines the pattern used to fill the inside of ovals, rectangles and polygons. The "N" item refers to "No fill", which is the default. |
| **Line Pattern** | ▤ | The line pattern palette sets the pattern used for all outer framing lines on objects. |
| **Fill Color** | ⬒ | The color palette is split into upper and lower halves which determine the colors used for the "foreground" (i.e. "black" areas of lines, text and fill patterns) and "background" (i.e. "white" areas of text and fill patterns). |
| **Line Width** | ⊟ | The line width palette determines the width of lines and object frames. |

## Placing Pins on a Symbol

The pin palette contains the tools needed to place connection pins on the symbol.



The top row of part pin tools are used to create pins on the part in any of the four orientations.

The second row of pins are provided as a convenience to create pins with an inversion bubble. A pin created with the inverted pin tools has no special characteristics, these are only provided as a graphical convenience.

The third row of pins are used to place bus pins on the part in any of the four orientations. Bus pins are described in more detail in “Placing a Bus Pin” on page 301.

When placing pins, a graphical pin is associated with a name in the pin list. The association is made by applying the following rules and using the first one that matches.

> Associate a selected pin name in the pin list which is unplaced and is of the same type. i.e.: normal and inverted pins can't be associated with bus pin names or internal bus pin names.

Associate an unplaced pin name of the correct type which follows the first selected pin name. Pin names are examined in a cyclic order so if the bottom of the pin list is reached the search continues from the beginning.

If no association is made then a new pin name of the correct type is created and added to the bottom of the list.

## Placing a Bus Pin

Bus pins allow busses to be connected to the part. A bus pin's functionality is determined by the internal pins it contains. These can be specified when the symbol is created, or modified later on the schematic using the Bus Pin Info command described in "Using Bus Pins" on page 214.

To add a bus pin to a symbol you need to perform two steps:

Use the Add Pins command to add the bus pin to the pin list and specify its bus internal pins. This command is described in "Adding Sequential Pin Names" on page 293.

Place the graphical bus pin using the techniques described in "Placing Pins on a Symbol" on page 300.

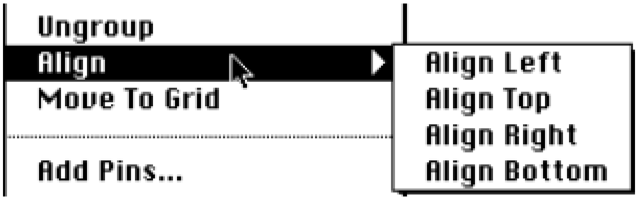
## Reordering Graphical Objects Front-To-Back

The Bring To Front and Send To Back commands are used to set the front-to-back ordering of the selected objects relative to the other graphic objects.

## Grouping Graphical Objects

The Group and Ungroup commands allow you to make multiple graphic objects, except pins, be treated as a single object or visa versa.

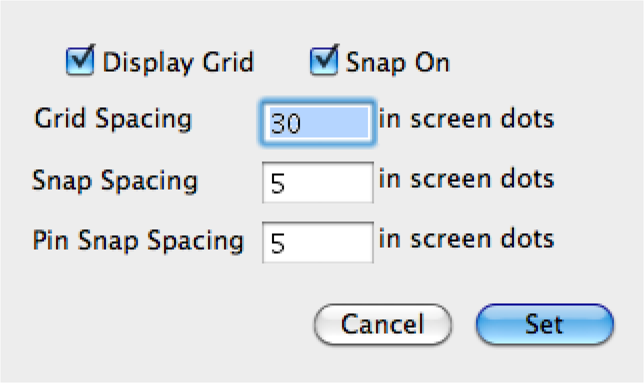
## Aligning Graphical Objects



The Align sub-menu allows you to pick how the selected objects will be aligned. For example, Align Left causes all of the selected objects to be moved such that their left edges are aligned with the left most selected object's left edge.

## Setting Grids

The Grids command allows you to specify the visible grid spacing and the snap-to grids for objects drawn using the drawing tools.



☑ Display Grid ☑ Snap On

Grid Spacing 30 in screen dots

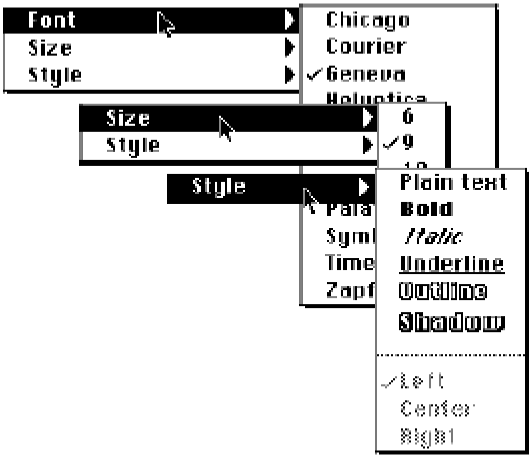
Snap Spacing 5 in screen dots

Pin Snap Spacing 5 in screen dots

| | |
|---|---|
| **Display Grid Checkbox** | This check box determines whether visible grid lines are shown in the drawing workspace of the DevEditor window. The spacing between these grid lines is determined by the value in the "Grid Spacing" field. |
| **Snap On Checkbox** | This check box determines whether the corners of objects made with the drawing tools are moved to the nearest snap-to grid point. |

| | |
|---|---|
| **Grid Spacing** | This number determines the spacing between the visible grid lines. The units are in screen dots at the default zoom level. |
| **Snap Spacing** | This number determines the spacing between snap-to points for the drawing tools (not including pins). This does not affect objects that have already been placed. The units are in screen dots at the default zoom level. |
| **Pin Snap Spacing** | This number determines the snap-to grid for device pins. This must be a multiple of 5 to meet the DesignWorks pin grid requirements. The units are in screen dots at the default zoom level. |

## Setting Text Font, Size and Style



These menu items set the text characteristics for the selected text objects. If no objects are selected when an item from a menus is selected then the selected text property becomes the new default.

# Entering Pin Information

## Selecting Items in the Pin List

Clicking on an item in the pin list selects that item and the associated

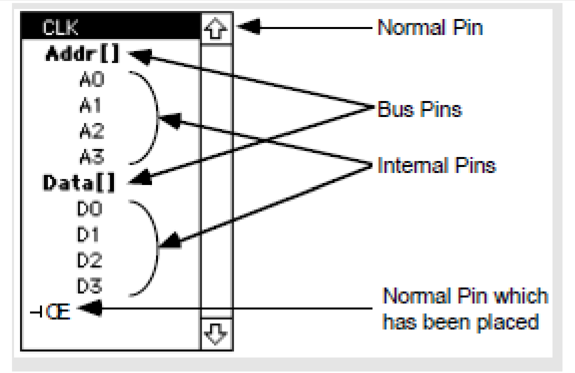
## DevEditor’s Pin List

The pin list box contains a scrollable list of the pin names associated with this device. This list is derived from the following sources:

> If the symbol was opened from an existing part, then the initial name list will be the pin names associated with the part.
>
> If a pin tool is clicked in the drawing workspace, a new name may be added with the form "PINxx" where xx is a sequential number. This is only done if no unplaced pin name could be used.
>
> If an open DesignWorks circuit is selected as an internal circuit, or a saved DesignWorks circuit is selected as an external circuit, then the names of the port connectors in the circuit will be merged with the names in the pin list.
>
> The Add Pins dialog can be used to create a list of pins to be merged with the pin names in the pin list.
>
> The Auto Create Symbol command can be used to specify all the pin names and pin numbers for a device symbol.

The pin name that appears in the pin list does not need to be the same as the graphical annotation which appears next to a graphical pin in the part's symbol. Some operations performed by the DevEditor - Auto Symbol Create - assume a correspondence but this is only done as a convenience. The result is that changing a pin name in the pin list will never cause the annotation next to a graphical pin to automatically change.

Any pin that has been placed (i.e. it has a corresponding graphical pin object which appears as part the symbol) will have a ⊣ mark beside it. Bus pins are marked in the list with "[ ]" following their name and the are displayed in bold. Internal pins (i.e.: pins which define the contents of a bus) are indented when displayed.



graphic in the drawing area, if any. The Shift key on the keyboard can be used to add to an existing selection, in the usual way. Most options in the Pin Info palette apply to multiple selected pins as well as single items.

While the Pin Info palette is displayed you can still click on items in the pin list to select them. In addition, pressing the Enter key moves to the next pin, SHIFT/Enter moves backwards one pin, Option/Enter moves the selection to the top of the pin list, Option/Shift/Enter moves the selection to the bottom of the list.

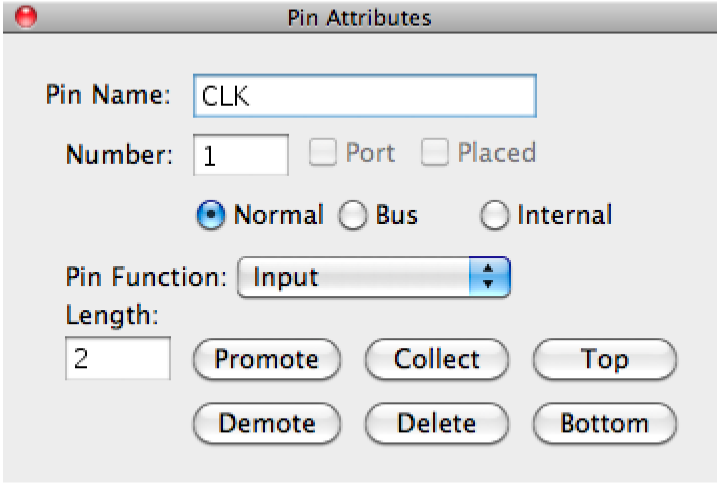
## Displaying the Pin Info Palette

To make the floating Pin Info palette appear, double-click on any pin name in the pin list. The window will display the state of the selected pins.

If a placed pin (i.e.: it has an associated graphical pin, and a "⊣" mark next to its name) is selected then both the pin and the graphical name will be selected. The opposite is also true. If a graphical pin is selected the associated item in the pin list will be highlighted as well. Internal pin names never have graphical associations, therefore selecting an internal pin name never selects a graphical pin.

The relationship between pin names in the pin list and graphical pins is not symmetrical. Every graphical pin must have a pin name, but every pin name does not necessarily have an associated graphical pin. This can lead to some surprises. For example,

> Selecting all graphical pins in the drawing workspace may not select all pin names: unplaced and internal pin names will not be selected.
>
> Deselecting all graphical pins in the drawing workspace may not deselect all pin names. Additional unplaced or internal names may have been initially selected.



Pin Name: CLK

Number: 1 ☐ Port ☐ Placed

● Normal ○ Bus ○ Internal

Pin Function: Input

Length: 2

The floating palette is divided into two parts. The top is used to examine and modify the characteristics of a pin. If more then one pin is selected in the pin list then only information that is common to all the selected pins will be displayed, but new values entered will be applied to all selected items.

The bottom of the Pin Info palette is used to modify the position and existence of pins in the pin list. All of the buttons in the lower half of the window affect all of the selected pins.

## Setting the Pin Name

The name field in the Pin Info palette has room for a 16 character identifier. The name must be unique within its context. The context for normal and bus pins is all other normal and bus pins. The context for a bus internal pin is all other internal pins in the same bus, except when the internal pin is not yet in a bus. In this case its context is the same as normal and bus pins. This means that regular pins and/or bus pins may not be the same, but that internal pins only need to be unique within their own bus.

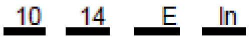
If only one pin is selected in the pin list then the name field can be changed. The change to the name will be applied when a new selection is made in the pin list or when a carriage return is typed. If the name is not acceptable for some reason a beep will occur and the name will not be changed.

If more then one pin is selected in the pin list then the name field will be empty the contents cannot be changed.

**TIP: If you are setting names on a number of pins, just hit Enter on the keyboard after each one to move to the next pin.**

## Setting the Pin Number

A four character identifier is displayed in the pin number field. Even though this is referred to as a number it may contain any valid character. For example, "10", "14", "E", and "In" are all valid pin numbers. The value entered for the number is displayed on the stem of a pin.



If more then one pin is selected in the pin list then the number field will show the number of the first pin selected, but the contents of the field cannot be changed.

## Selecting Normal/Bus/Bus Internal Status

The three radio buttons marked "Normal", "Bus", and "Bus Internal" show the current type of the pin. If the pin is not associated with a graphic pin then the radio buttons are enabled and may be changed. Changing the radio buttons causes the pin name to be displayed differently in the pin list. If the pin name is associated with a graphic pin then the radio buttons will not be enabled but will display the current state of the pin.

If more then one pin name is selected in the pin list then the buttons will show a current setting only if all selected items are the same. In any case, selecting a new setting will affect all selected pins.

## Setting the Pin Type

The pin type (e.g. input, output, etc.) is selected using the Pin Function pop-up menu. The pop-up will only be displayed for pins of type "Normal" or "Bus Internal". Pins of type "Bus" do not have their own type.

Using the pop-up menu the following functions may be assigned to the pin: Input, Output, Tristate, Bidirectional, Open Collector, Open Emitter, Tied High, Tied Low, Latched Input, Latched Output, Clocked Input, Clocked Output, Clock Input, and No Connect.

The pin type is important to simulation - is this pin driving or driven, and additionally can be important in error checking and report and netlist generation.

***See Appendix E—Device Pin Types on page 385 for more information on pin types.***

You can apply a new pin type setting to any number of selected pins. If more then one pin is selected in the pin list then the function pop-up menu will show a current value only if all selected items are the same. In any case, selecting a new setting will affect all selected pins.

## Port Status Indicator

The port checkbox cannot be changed by the user and is only used for information. When checked it indicates that the pin name matches a port name in an attached subcircuit. The box will be unchecked for any of these reasons:

No attached internal or external subcircuit.

No port with the same name in the subcircuit.

The pin is an internal pin and the pin's bus name is not the same as the bus port in the subcircuit.

## Placed Status Indicator

The placed check box cannot be changed by the user and is only used for information. When checked it indicates that the selected item in the pin list is associated with a placed graphical pin.

## Reordering Pins in the Pin List

A number of controls are available in the Pin Info palette to change the order in which pins appear in the pin list. This pin order does not affect the graphical appearance of the symbol, but may affect netlists generated from schematics containing the symbol. For example, the SPICE netlist format depends upon the device pin order matching the order expected by the target simulator.

***For more information on pin order and how it affects interfaces to other systems, see Chapter 6—Before Starting a Major Design on page 117.***

These controls are available to change the pin order:

> The Top and Bottom buttons move the selected pins in the pin list to the top or bottom of the pin list, respectfully.
>
> The Promote and Demote buttons move the selected pins in the pin list up or down one pin in the pin list, respectfully. If a selected pin reaches either the top of the list when promoting, or the bottom of the list when demoting, then it will stop and the remaining selected pins will begin to collect at the top or bottom of the list, respectfully.
>
> The Collect button causes all of the selected pins in the pin list to be grouped following the first selected item.

## Creating a Part With a Subcircuit

### Creating the Port Interface

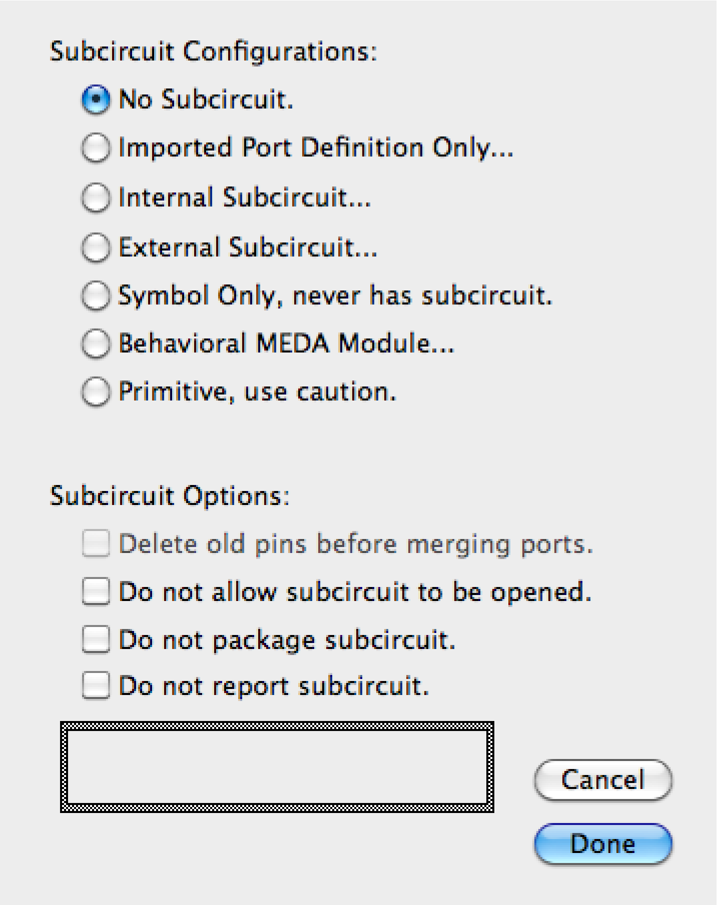
A part's subcircuit is a schematic circuit which is associated with the part such that it is considered to be the part contents. A circuit which is to be used as a subcircuit must include parts called port connectors. Port connectors, which are named, allow signals in the circuit to be associated with pins on an enclosing part. Port connectors make this association by name, i.e.: a port connector named "A0" will only associate with a pin with the same name.

***See "Creating and Using Port Connectors" on page 263 for more information on the port interface.***

### Selecting the Subcircuit

The easiest method to create a part with an associated subcircuit is to begin by selecting the schematic circuit to be used as the subcircuit. This sounds like stating the obvious, but creating the part in any other order is more work.

To select the circuit to be used as the part's subcircuit you must also define the type of part being made. Select the Subcircuit & Part Type command from the DevEditor menu.



Subcircuit Configurations:

● No Subcircuit.

○ Imported Port Definition Only...

○ Internal Subcircuit...

○ External Subcircuit...

○ Symbol Only, never has subcircuit.

○ Behavioral MEDA Module...

○ Primitive, use caution.

Subcircuit Options:

☐ Delete old pins before merging ports.

☐ Do not allow subcircuit to be opened.

☐ Do not package subcircuit.

☐ Do not report subcircuit.

This dialog allows you to select among several different options:

| | |
|---|---|
| **No Subcircuit** | This is the default. It indicates that the part being made has no associated subcircuit, but it doesn't rule out that a subcircuit may be attached by the schematic capture section of DesignWorks. |

| | |
|---|---|
| **Imported Port Definition Only** | This is similar to the No Subcircuit option. No subcircuit becomes associated with the part, but a port interface is extracted from a circuit for use with the part. Selecting this option prompts you to pick an open schematic circuit. The port names are read from the circuit without attaching the circuit to the part. Like the above option, this doesn't stop a subcircuit from being attached later. Since a subcircuit wasn't attached to the part, the port interface is only associated with the part for the length of the editing session. |
| **Internal/External Subcircuit** | These two options prompt for an open circuit and a saved circuit, respectfully. The selected circuit becomes associated with the part being created as its subcircuit. The difference is that the Internal Subcircuit option includes the circuit definition as a piece of the part and saves that piece with the part in the library. The External Subcircuit option only includes a file reference to the circuit file with the part in the library. Both options import the circuit's port interface so that the names of the ports are available in the DevEditor. |
| **Symbol Only** | This option is like the No Subcircuit option except its doesn't allow the schematic capture part of DesignWorks to associate a subcircuit with the part in the future. |
| **Behavioral & Primitive** | (See their related section of this manual). |

There are two subcircuit options in the lower part of the dialog:

| | |
|---|---|
| **Delete old pins before merging ports** | This option is only enabled when the selection made above results in a port interface being read. If this box is checked when the "Done" button is pressed then all of the old pins in the pin list will be deleted. This allows a new port interface to be brought in without any conflicts with the existing pin list. If this option wasn't checked then port names would be merged with the names in the list. Duplicate pin names and their related properties remain unchanged, except they are now associated with the new port. Unmatched pin names in the pin list remain exactly the same. |

| | |
|---|---|
| **Protect part from being opened** | This option has the effect of saying, "Yes. There is a subcircuit, but in general you don't want to go into it". This causes the schematic capture part of DesignWorks to prompt to make sure it is really OK to enter the subcircuit before doing so. It also controls if the report generator will list the contents of this device in a netlist. In general this is used for parts like 7400 series devices. Where a designer treats the part as a physical component, but there may be a subcircuit for the simulation purposes. |

If option 2, 3, or 4 was selected in the dialog then upon exiting from the dialog a port interface will be read and the pin list in the DevEditor window will be updated to show the new pin names. In addition, if option 3 or 4 was selected then a subcircuit is also attached to the part being created in the DevEditor window.

## Drawing the Graphics

The graphic image of the part may be drawn in the same way as was described in the previous section on creating part's without subcircuits.

***You can also generate a symbol automatically using the Auto Create Symbol command. See the section Auto Creating a Symbol.***

## Associating Graphic Pins with Pin Names

Pins in the pin list that may be placed are those that are not indented and that don't have a symbolic pin symbol next to their names. Names that have a symbolic pin next to them already have a graphical association. Names which are indented are internal bus pins and don't have a graphical representation.

Assignment of graphical positions to pins may be done explicitly by using the pin placement tools or automatically using the Auto Create Symbol command. See Auto Creating a Symbol, below.

To explicitly assign a graphical pin position to a pin requires that the pin be selected in the list and that a graphical pin be placed using one of the pin tools.

A graphical pin assignment will only be made with an unplaced pins of the correct type. If a placed pin is selected in the pin list, and an attempt is made to associate a graphical pin with it, then the association will not be made with the pin because it was already placed. Instead the new graphical pin will be associated with the next free pin of the correct type or a new pin will be created if there are no unplaced pins of the correct type.

Because of the way the pin association mechanism works, i.e.: searching for the next placeable pin, it is possible to place all pins of the same type without having to make a new selection. Select the first unplaced pin and place it. Now since the current selection is placed, the next graphic pin placed will be matched with the next pin of the correct type. Place another pin and note how the selection moves forward to the next previously unplaced pin of the same type.

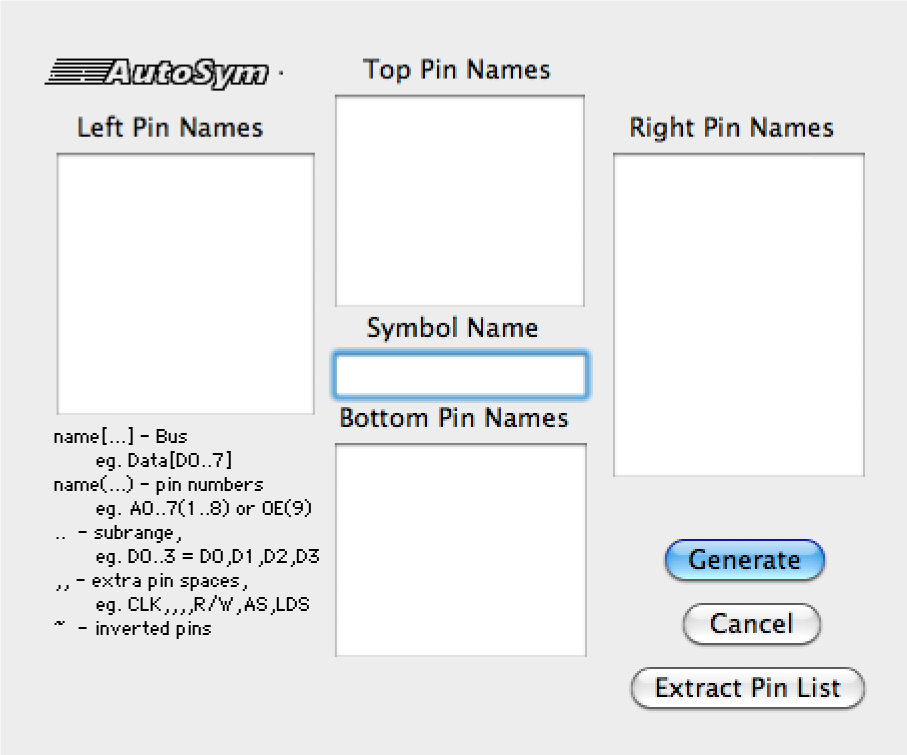
## Automatically Creating Symbols

Two mechanisms are available in DesignWorks for automatically generating device symbols for standard types of parts. One is the Auto Create Symbol command in the DevEditor tool. This command will generate a standard, rectangular symbol given a list of names for the pins on each side of the symbol. The other mechanism is the ConnMaker tool, which will generate a variety of connector symbols with a given number of pins. These tools are described in the following sections.

### Auto-creating Rectangular Symbols in the DevEditor

The Auto Create Symbol command will create standard rectangular symbols given a list of the desired input and output pin names. For maximum flexibility, the symbol generated consists of separate graphic objects and is completely editable after it is generated.

The current settings for line width, fill patterns, color, text font, size and style, etc. are used in generating the symbol. The only exception to this is the type name text placed in the center of the symbol, which is written 3 points larger than the current setting and in bold.



```
name[...] - Bus
      eg. Data[D0..7]
name(...) - pin numbers
      eg. A0..7(1..8) or OE(9)
..  - subrange,
      eg. D0..3 = D0,D1,D2,D3
,, - extra pin spaces,
      eg. CLK,,,,R/W,AS,LDS
~  - inverted pins
```

Selecting the Auto Create Symbol command displays the above dialog box. The pin name boxes will contain the information entered the last time the DevEditor was invoked for this part. These can be modified as desired. The new settings will be merged with pin list when the Generate button is pressed.

**NOTE: All pins on the generated symbol will have a pin type of "input", regardless of position on the symbol. If this is not appropriate for your application, it must be changed manually for the pins requiring a different setting. See "Setting the Pin Type" on page 307 for information on setting pin type.**

### Entering Pin Names

The four pin name boxes allow you to specify the names of pins to appear on the left, right, top and bottom of the device symbol. The syntax described in the previous section, Adding Pin Names, is also used to define the pin names and numbers with the following extensions:

> Inverted pins can be specified using the "~" character in front of the name. The inverted pin will be displayed with a bubble. The "~" will not appear in the symbol.
>
> Items in a list can be separated by blanks or commas. Placing an extra commas between two items adds extra space between the pins on the symbol. Additional space can be added with more commas.

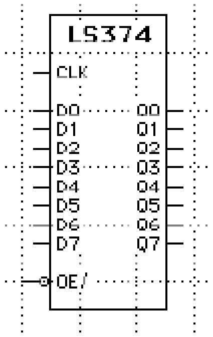
### Entering the Symbol Name

The symbol name text box allows you to specify the text that will appear centered at the top of the symbol. This also becomes the new name for the part.

### Generate Button

The Generate button causes the current contents of the active drawing window to be erased and replaced by the generated symbol. The pin list will be merged with the new pins described in the dialog. This symbol consists of standard graphic objects so it can be edited using any of the drawing tools provided.

An example of a device produced by the Auto Create Symbol:



### Using Pins Already in the List

The Extract Pin List button updates the pin name boxes with the names extracted from the pin list. Pins which are inputs and busses are placed on the left, outputs are placed on the right.

## Auto-creating Connector Symbols

The ConnMaker tool allows you to easily generate a symbol for a connector with any number of pins. ConnMaker provides a selection of common graphics for the connector pins and a number of symbol options. The generated symbol is written to a symbol library of your choice, from which you can place the connector on your schematic or use the DevEditor to perform custom editing or make attribute settings appropriate for your application.

To generate a connector symbol:

Select the ConnMaker tool in the Tools menu.

Enter the name you wish the symbol to be saved under in the library. You will select the library to save to later.

Enter the number of pins needed.

Change the pin spacing, if desired. The spacing number is entered in grid units, i.e. multiples of 5 screen dots at normal zoom.

Select the pin graphic type by clicking on the button next to the desired graphic. There is no ability in this version of ConnMaker to provide your own pin graphics.

Select the "Draw Frame" option if you wish to have a rectangular frame included in the symbol.

Select the "Assign Default Pin Numbers" option if you wish sequential pin numbers to be assigned starting at 1. If you need some other pin numbering scheme, leave this option off and edit the symbol using the DevEditor after saving it to a library.

Click the Create button. A list box will appear displaying all currently open symbol libraries, with options to create a new one or open a library file from disk.

Select the desired destination library and click on the Save button.

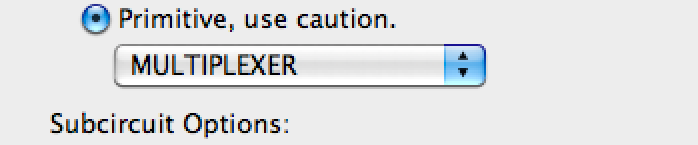
The generated symbol will be written to the selected library and can now be used on the schematic or editing in the usual way.

# Creating Special-Purpose Symbols

## Assigning a Primitive Type

**NOTE: The Primitive type settings should only be used with a clear understanding of their functions. Primitive types are intended primarily for use with the DesignWorks Digital Simulation option. See Appendix C—Primitive Device Types on page 381.**

The primitive type of a part allows DesignWorks to recognize a number of special device types such as pseudo-devices and simulation primitives used by the DesignWorks Simulator option.


● Primitive, use caution.

MULTIPLEXER

Subcircuit Options:

To select the primitive type choose the Subcircuit & Part Type command in the DevEditor menu. Then select the "Primitive" radio button. This will cause a pop-up menu to be displayed below it. The pop-up menu will show the primitive type currently selected. Clicking on the pop-up menu will display the other options available. The following manual sections describe the usage of some of these settings. See Appendix C—Primitive Device Types on page 381 for a complete list of primitive types.

## Creating Primitive Devices for use with the Simulator

Device symbols for use as simulation primitives with the DesignWorks Simulator must have very specific primitive type and pin type settings and pin orders. Refer to the DesignWorks Simulator Reference Manual for complete information.

## Creating a Breakout

A breakout is a special device that allows signals to be associated with a bus. It consists of 1 Bus pin, no internal pins, and N normal pins.

Because the device type is breakout the normal pins will be connected to like named signals in the bus. Breakouts are normally created using the New Breakout command described in “Using Bus Breakouts” on page 211.

**NOTE: Another way to split signals out of a bus is to use a part with a subcircuit as a splitter, which explictly routes signals between pins. This has advantage of flexibility since signals do not have to be explictly broken out but may instead be split into busses or any combination of busses and signals. The disadvantage is that since the splitter is a device it will be listed in hierarchical netlists. In flattened netlists it will not appear if marked as a non-protected device. In addition, there is a significant memory and file size penalty.**

## Creating a Power (Signal) Connector

A power or signal connector is a special type of device which is generally used to represent a power or ground source, e.g.: +5V, +15V, -15V, -5V, Ground, Vss, Vdd.

These devices have a special properties in the schematic. When a pin on one of these devices is connected to a signal it attempts to assign its pin name to the signal. If the signal doesn't have a name then it gets the name of the pin. If the signal is named, and the name is different from the pin's name, then you will be prompted to select between the signal name and the pin name.

An additional property of signal connectors is that any signal they are connected to is exported across all pages of the schematic.

The type of the pin must be set correctly if simulations are to make sense. For signal sources like +5V, +15V, and Vss, a normal simulation would expect a logical value of 1 (True). For signal sources like Ground and Vdd, a logical value of 0 (False) is expected. This means the pin type should be "Tied High" or "Tied Low" (See “Setting the Pin Type” on page 307).

If you don't want the signal connector to supply a signal value, but only its name and the fact that it makes the signal global, then the pin type should be set to "Input".

## Creating a Port Connector

Port connectors have the property of associating a signal or bus with a pin on an enclosing part. The association is made by name. The port's Name is compared to the pin names on the parent part. Internal pins in busses are matched by pin name.

***See "Creating and Using Port Connectors" on page 263 for more information on port connectors.***

### Creating a Signal Port Connector

A port connector for a signal must have a pin which is of the correct type to interface the signal to the parent part's pin. For example, an input pin on the port connector would be correct to connect to a output pin on the parent part. The name of the port connector pin is not important. Only the name assigned to the port connector once it is placed is important; it must be the same as the parents pin.

### Creating a Bus Port Connector

A port connector for a bus must have a bus pin which contains pins of the correct type. For example, a bus pin with three internal pins A (input), B (input), and C (output) would be correct to connect to a parent part's bus pin which contained pins A (output), B (output), C (input). The name of the port connector bus pin is not important, but the internal pins must have the same name. Once the port connector is placed in a circuit its reference name is important, it must be the same as the parent's pin name.

## Creating a Page Connector

Page connectors have the property of exporting the signal allowing it to be connected to across all pages in a circuit.

***See "Inter-page Connections" on page 219 for more information.***

### Making a Signal Page Connector

The graphical appearance of a signal page connector is not important. The only characteristics it must have are:

It must have only a single input pin. The name of the pin is not important.

It’s primitive type must be set to Page Connector using the DevEditor’s Subcircuit and Part Type command.

### Making a Bus Page Connector

A page connector for a bus is the same as a signal page connector except that is has only a single bus pin with no internal pins.

## Features Requiring Symbol Attributes

A number of DesignWorks features rely on certain attribute fields being present in a part definition. If you create a part symbol with no default attributes it will have the following characteristics:

The Packager will assume that it has only one unit per package, i.e. a unique package name will be assigned to each device and the Unit field will be left empty.

If you have not assigned default pin numbers to the pins, they will have to be entered manually for each device on the diagram.

Part number, component value and package attributes, if needed (e.g. for PCB netlists), will have to be entered manually for each device on the schematic.

No power and ground connections will be made automatically.

The default prefix will be used to auto-generate device name when the part is placed.

The following sections describe the attribute entries that should be made in each symbol you create in order to take advantage of these DesignWorks features.

### Gate Packaging

A number of attribute fields are used by the Packager when it assigns a name and unit to a device. If you intend to create your own library symbols that require gate packaging, you will need to be familiar with these fields. If you are only using the libraries provided with DesignWorks, or if you have no requirement for gate packaging, you will not need this information.

***See a more detailed description of the packaging mechanism and fields in "Creating a Symbol with Multi-gate Packaging" on page 142.***

## Auto-Naming

Both the Packager and the Auto-Naming features make use of attribute fields in the generation and positioning of names for devices.

**NOTE: DesignWorks allows you to select any device field to be used as the prefix for default name generation. The prefix fields described here are the standard ones included in the DesignWorks libraries, but you can also define your own if you have special naming requirements. See the description of the PrefixField design attribute elsewhere in this manual.**

| Field Name | Set In | Description |
|---|---|---|
| Name.Prefix | Device | This field gives the prefix to be used in generating a device name or package name. In the DesignWorks standard libraries, this field is empty for most integrated circuit parts. They therefore take the default prefix specified with the design. For discretes, the common prefix character is specified. |
| Name.Spice | Device | This field is specified only for symbols to be used with a SPICE-based analog simulator. It is an optional name prefix matching the standard required for SPICE models. |
| Name.Pt | Device | This field contains an X,Y coordinate for the default position of the name when the device is placed. This is specified in 1/1000 inch and is relative to the top left corner of the symbol. E.g. "1000, -200". See "Using Default Position Fields" on page 190 for more options in this field. |

***See "Setting the Auto-Generated Name Format" on page 136 for more information.***

## Specifying Part and Package Type Information

If the netlist output from DesignWorks is to be used to generate a printed circuit board, then it will be necessary to associate package type information with each device. This is done by specifying a keyword for each package type in the Package attribute field. Package type fields are included in the standard type libraries, but may be overridden if necessary by specifying a value for the Package field in the device attributes.

The keywords associated with the common package types are described in a technical note supplied with the package.

**NOTE: The package code entered in the Package attribute field is used only to pass information to an external PCB layout package. It is not used internally by DesignWorks for gate packaging or any other operation.**

***See Chapter 6—Before Starting a Major Design on page 117 for more information on passing data to an external PCB package and some other alternative ways of specifying data.***

### Package Types for Discrete Components

Discrete components do not normally have standard package types. If a package type is required for PCB layout purposes, then the "Package" field in the attributes must be set manually for each device.

### Part and Package Fields

The following fields are the standard ones used to specify the part number and package type for netlisting purposes.

**NOTE: The standard part and package fields are listed here. You are free to add new fields for special purposes at any time and modify the field names used in the netlist formats.**

| Field Name | Set In | Description |
|---|---|---|
| Part | Device | This field gives the part name that will be used for netlisting purposes. This may be the same as the name as the part is saved under in the library, or it may be in a format required by some external package. |
| Part.List | Device | This field contains an optional list of part numbers that can be represented by this symbol. If specified, this list will appear in a value list menu in the device pop-up menu on the schematic. |

| | | |
|---|---|---|
| Part.Pt | Device | This field contains an X,Y coordinate for the default position of the Part field when the device is placed. This is specified in 1/1000 inch and is relative to the top left corner of the symbol. E.g. "1000, -200". See Chapter 5 - Advanced Schematic Editing for more options in this field. |
| Package | Device | This field specifies the default package type for use by an external PCB layout package. |
| Package.List | Device | This field contains an optional list of package codes for the device. IMPORTANT: This field is unusual in that it is linked in DesignWorks to the Part.List field. If you select an item in Part.List, the corresponding item in Package.List is automatically placed in the Package field. |

***See Chapter 6—Before Starting a Major Design on page 117 for more information on passing data to an external PCB package and some other alternative ways of specifying data.***

## Fields for Hierarchy Operations

The Restrict attribute field is used to determine whether packaging, report generation and editing operations are allowed to proceed down into the sub-circuit block. The allowable values for this field are discussed in Appendix A—Predefined Attribute Fields on page 373.

## Other Fields

The following fields are included in the standard DesignWorks libraries and are frequently used to pass data to external packages in netlists and reports. However, these fields are not required for any specific program features.

***See the Read Me file provided with the design kit you are using for specific requirements of the netlists and reports you are planning to use. In addition, you may wisth to look at Chapter 6—Before Starting a Major Design on page 117 for an overview of attribute usage and how it may affect interfaces to other systems.***

| Field Name | Set In | Description |
|---|---|---|
| Power | Device | This field contains a list of pins to be connected to the power net in netlist reports. See “Power and Ground Connections in Attributes” on page 225 for more information. |

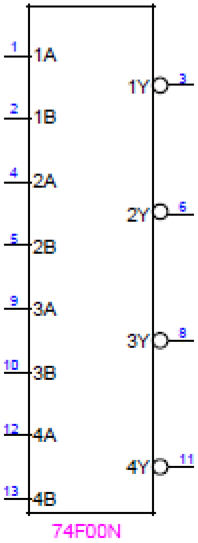
| | | |
|---|---|---|
| Ground | Device | This field contains a list of pins to be connected to the ground net in netlist reports. See “Power and Ground Connections in Attributes” on page 225. |
| Function | | This field contains a terse symbol function code. A full list of the codes used in the standard DesignWorks libraries is given in “Function and Category Codes” on page 333. |
| Category | Device | This field contains a terse symbol category code. A full list of the codes used in the standard DesignWorks libraries is given in “Function and Category Codes” on page 333. |
| Description | Device | This type attribute contains a short description of the symbol type. For instance "Quadruple 2-Input Positive-NAND Gates". |
| Permutable | Device | This field contains a list of the swappable pins and gates in this package and is intended for external use by PCB packages. For more information on the format of the Permutable attribute see “The Permutable Attribute” on page 329. Note that this field is not used internally for packaging operations. For the packaging fields, see “Setting Packaging Attribute Fields While Creating a Symbol” on page 143. |
| Value | Device | This field contains the default component value for the part. It normally applies only to discrete components. |
| UnusedPins | Device | This field contains a list of the unused pins in this part type. This field is not included in any of the standard DesignWorks report formats, but is included for general usage. |

## Using the Standard DesignWorks Libraries

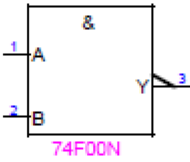
The libraries provided with DesignWorks 3.0 contain over 12,000 symbols including both ANSI and IEEE format symbols. The purpose of this manual is to describe the naming, graphical, attribute and organization conventions used in the libraries. This information will be of interest if you if you need to adhere to some particular drawing standard or if you plan to export data from DesignWorks to another system.

## Symbol Format

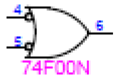
The DesignWorks libraries include the ANSI and IEEE symbol equivalents for each part. The installation procedure allows you to choose to install only one symbol type.



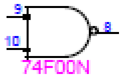
ANSI-Monolith Style



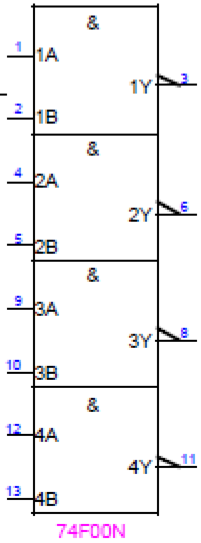
IEEE Style



Demorgans Style



ANSI Style



IEEE-Monolith Style

## Finding a Library

***See also the SearchLib manual for information on searching for a specific part by characteristics.***

The directory structure used to store the libraries is divided into several folders. The top most folder is separated into Analog, ANSI and IEEE.

Within Analog folder you will find:

Connectors

Diodes

Discretes

National Linear

Transistors

Spice Lib

Misc. Analog

Within the ANSI and IEEE folders you will find:

54 TTL

74 TTL

75 TTL

CMOS

ECL

Memory

Micro Processors

Miscellaneous

PLD

The 54, 74 and 75 series components are divided into their families, so there are separate libraries for the expanded components in the folder "Expanded" and non-expanded components in the "Monolith" folder. Within the Expanded folder where appropriate there is also a "DeMorgans" folder that contains the DeMorgan equivalent symbols for many of the components. All the TTL components are derived from Texas Instruments data.

Within the TTL folders are the following library files:

74, 74S, 74LS, 74AS, 74ALS, 74AC, 74ACT, 74F, 74HC, 74HCT, 74HCU.

54, 54S, 54LS, 54AS, 54ALS, 54AC, 54ACT, 54F, 54HC, 54HCT, 54HCU.

75ALS.

Memories are divided into Motorola, Texas Instruments (TI) and Toshiba.

Micro Processors are divided into Intel, Motorola (Freescale) and Zilog.

## Interpreting Library Part Names

A component name is made of up to five parts.

The Logic Family Name

The Device Number

Device Attributes

DesignWorks unit for packaging

A marker indicating the symbol style

Most of these name parts are only used for the TTL family of parts. Other components may have just the manufacturer part name or a descriptive name as the library name, for instance "DIODE" or "2N222".

Some parts, like diodes, have over 100 different names in the parts list. In this case the library part name will be the first name that appears in the Part.List attribute.

Occasionally the part name is too long to display in the parts palette list, in this case the name will appear truncated.

### The Logic Family Name

The logic family name is usually from one of the following:

74, 74S, 74LS, 74AS, 74ALS, 74AC, 74ACT, 74F, 74HC, 74HCT, 74HCU.

54, 54S, 54LS, 54AS, 54ALS, 54AC, 54ACT, 54F, 54HC, 54HCT, 54HCU.

75ALS.

### The Device Number

The device number is the middle digits in the component name and corresponds to the symbol type. For example "123" in 74ALS123J.

### The Device Attributes

The device attributes are the manufacturer attributes that apply to this symbol. The attributes are enclosed in parentheses "()" such as (W,J).

### Unit for Packaging

The presence of a unit indicates that there are multiple symbols required to represent all the units in this package. The unit is appended to the name with a "." so ".a" would indicate that this component will default to unit "a".

If there are different symbols for some of the units within a package then the different units will each have their own entry in the library. The unit letter is appended to the end of the library name such as: "5450(J).a" and "5450(J).b". This component is a Dual 2-Wide 2-Input AND-OR-INVERT Gate (One Gate Expandable). In some cases the part that shows up as xxx.a will have a "c" and "d" units and the "b" unit will be a different entry because it does not share the same symbol.

### Symbol Style

The symbol style marker is one of the following:

| | |
|---|---|
| **A** | ANSI style symbol. |
| **I** | IEEE style symbol. |
| **D** | DeMorgan equivalent style symbol. |
| **AM** | ANSI Monolith (non-expanded gates). |
| **IM** | IEEE Monolith (non-expanded gates). |

### Library Name Example 1

A 54S00J may be listed in the library selection palette as "54S00(W,J)A"

"54S" is the TTL family name, this part of a library name is only used for the TTL components.

"00" is the component number

"(W,J)" indicates that there are two part suffixes for the 54S00 a "54S00J" and a "54S00W".

The "A" suffix indicates that this part is an ANSI style symbol.

### Library Name Example 2

A 5450J may be listed in the parts palette as "5450(W,J).aI".

"54" is the TTL family name, this part of a library name is only used for the TTL components.

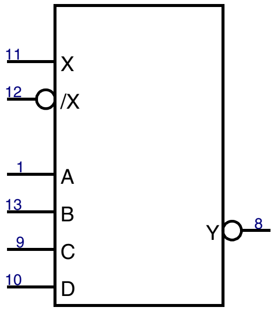
"50" is the component number

"(W,J)" indicates that there are two part suffixes for the 5450 a "5450J" and a "5450W".

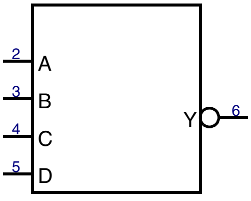
There may be a ".a" (or ".b", ".c" etc…) after the part suffix as in

"5450(J).a". This is the unit and it indicates that there is more than one symbol for the gates in a package. Most of the TTL gates have multiple units in a package, the unit name only appears in the library if the graphic symbol for one or more of the units differs from the others.

The "I" suffix indicates that this part is an IEEE style symbol.



5450J.a



5450J.b

## The Permutable Attribute

The permutability of pins on a device will be indicated by an attribute in the symbol definition. This field has the following characteristics:

The attribute used is "Permutable".

The value of the Permutable attribute is a string of bracketed pin numbers.

There are two types of brackets:

'(' ')' enclosing a list of permutable pins.

'[' ']' enclosing a list of non permutable pins.

Pin lists are separated by ',' characters.

Pins that are not in the list are not permutable.

The permutable and non-permutable brackets are nested to indicate conditions where individual pins are not permutable but as a group they may be.

A list of pins inside a () are permutable. e.g. (1,2,3).

A list of pins inside a [] are not permutable. e.g. [1,2,3].

Lists of non permutable pins [1,2,3] inside a () indicates that the individual non permutable pins may be permuted as a group. e.g. ([1,2,3],[4,5,6])

Complex permutations as in Example 3 below may indicate that within a non permutable unit some of the pins may be permuted. Such as the A and B inputs on a NAND gate e.g. (1,2) but the output Y must not be swapped with another pin without swapping both A and B. e.g. [(1,2),3] so we get a list like ([(1,2),3],[(4,5),6]).

This means pins 1 and 2 may be swapped (they are functionally identical). Pins 4 and 5 may be swapped. If we swap pin 3 or 6 we must swap 1,2,3 or 4,5,6 at once.

### Example 1

A 54LS74 has two D type flip flop units each identical. To permute one pin we must permute all pins on one FF with the ones on the other FF. This is indicated as follows:

Permutable = "([14,1,2,3,13,12],[8,7,6,5,9,10])";

pins 14,1,2,3,13,12 are non permutable with each other.

pins 8,7,6,5,9,10 are not permutable with each other.

But as a group pins [14,1,2,3,13,12] are permutable with pins [8,7,6,5,9,10].

### Example 2

A 54LS04 Hex Inverter. The two pins on each device are not permutable since one is an input and the other is an output. But each of the six inverters can be exchanged as a whole with another.

Permutable = "([1,14],[3,2],[5,6],[7,8],[9,10],[13,12])";

### Example 3

A 54LS00 Quad 2-Input Nand Gate. The 4 individual units of the NAND may be permuted as a whole as well the two inputs of each may be permuted. This is indicated as follows:

Permutable = "([(1,2),3],[(6,7),5],[(9,10),8],[(12,13),14])";

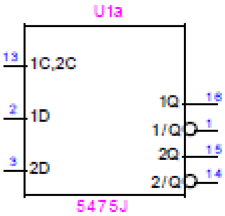
In this example each non permutable group "[(1,2),3]" is one of the NAND gate units for this unit pins 1 and 2 are permutable. Pin 3 is not.

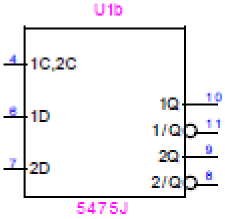
However each unit may be permuted as a whole.

### Example 4

A 5475J 4-Bit Bistable Latch. There are 4 nearly identical latches in this package. Each has a D input, a Q and \Q output and an enable (C). However there are only 2 enables for the 4 bits. Bits 1 and 2 have a common enable (Enable 1-2) and bits 3 and 4 have a common enable (Enable 3-4). This differentiates bits 1 and 2 from bits 3 and 4. The D,Q,\Q pins on bit 1 may be exchanged with the same pins on bit 2, but if we exchange bit 1 for bit 3 or 4 we must also exchange the Enables. This is indicated as follows:

    ([13,([2,16,1],[3,15,14])],[4,([6,10,11],[7,9,8])])





## Package Codes

Package codes have three parts:

Manufacturer Name

Number of Pins

Package Style code

The manufacturer name may be any string of characters although it is advised that no numeric digits should be used since that may confuse the format, the first digits in the package name should be the number of pins used in the package.

### Example

TI14DPN

This package is manufactured by Texas Instruments (TI) there are 14 pins on the device and the style is a narrow, plastic dual in line package.

In general the physical dimensions of the packages are not defined by these codes. The DIP package types are standard, others may vary with the manufacturer. The package codes define the sets of foot prints required for PCB layout, it is up to the PCB layout tool to define the actual physical dimensions for the package foot print.

The Package styles used in the DesignWorks libraries are as follows:

| **Package Type** | **Code** |
|---|---|
| DIP, plastic narrow | DPN |
| DIP, plastic medium | DPM |
| DIP, plastic wide | DPW |
| DIP, plastic x-wide | DPX |
| DIP, ceramic narrow | DCN |
| DIP, ceramic medium | DCM |
| DIP, ceramic wide | DCW |
| DIP, ceramic x-wide | DPX |
| DIP, side brazed narrow | DBN |
| DIP, side brazed medium | DBM |
| DIP, side brazed wide | DBW |
| DIP, side brazed x-wide | DBX |
| DIP, windowed narrow | DVN |
| DIP, windowed medium | DVM |
| DIP, windowed wide | DVW |
| DIP, windowed x-wide | DVX |
| MODULE - Leadless/single side | M |
| MODULE - Leadless/double side | MD |
| MODULE - Single Sided with Edge Clipped Leads | E |
| MODULE - Double Sided with Edge Clipped Leads | ED |
| MODULE - Single Sided with Pin Centered | I |
| MODULE - Double Sided with Pin Centered | ID |
| ZIP | Z |
| Flatpack | F |

| | |
|---|---|
| Flatpack, windowed | FV |
| Flatpack, Sidebrazed | FB |
| Quad Flatpack | Q |
| SOIC narrow | SN |
| SOIC medium | SM |
| SOIC wide | SW |
| SOIC x-wide | SX |
| Quad SOIC, Gullwing | G |
| Quad SOIC, Gullwing/Rectangular | GR |
| SOJ (J-bend SOIC) | J |
| 20/26 pin SOJ | J |
| SOIC, ceramic | JC |
| Pin Grid Array | A |
| Pin Grid Array, windowed | AV |
| Pin Grid Array, plastic | AP |
| Pin Grid Array, cavity down | AD |
| Pin Grid Array, with Heatsink | AH |
| Leadless chip carrier, ceramic Square | L |
| Leadless chip carrier, ceramic Rectangular | LR |
| Leadless chip carrier, sidebrazed Square | LB |
| Leadless chip carrier, sidebrazed Rectangular | LBR |
| Leadless chip carrier, windowed Square | LV |
| Leadless chip carrier, windowed Rectangular | LVR |
| Leadless chip carrier, plastic Square | P |
| Leadless chip carrier, plastic Rectangular | PR |
| Leadless chip carrier, ceramic Square | C |
| Leadless chip carrier, ceramic Rectangular | CR |
| Leadless chip carrier, ceramic windowed Square | CV |
| Leadless chip carrier, ceramic windowed Rectangular | CVR |
| Leadless chip carrier, plastic with bizarre attributes | PQ |

## Function and Category Codes

The following table lists the Category and Function values used in the standard DesignWorks libraries. Other values may of course be added as libraries are augmented in future releases.

| **Category** | **Function** |
|---|---|
| ARITH | ACCUM, CARRY, ALU, MULT, ADD |
| BUFF | BUFFINV, DELAY, BUFF, INV |
| COMP | ADDCOMP, ERRDET, PARITY, COMP |

| | |
|---|---|
| CONN | CONNM, CONNF |
| CTR | PRESCALE, TIMER, OSC, FREQDIV, RATEMULT, CTR |
| DIODE | TUNNEL, LED, BRIDGE, ZENER, DIODE |
| ELMECH | RELAY, SWITCH |
| FF | DFF, JKFF |
| GATE | OAI, XNOR, GATE, XOR, AOI, OR, AND, NOR, NAND |
| LATCH | MULTIVIB, LATCH |
| LIN | ADCONV, VOLTREG, LIN, OPAMP |
| MEM | ROM, EPROM, EEPROM, PROM, DRAM, FIFO, SRAM |
| MISC | LVLCNV, PLL, RCVR, MISC, FILTER, DSWITCH, PLSSYNCH |
| MUX | AMUX, BITSHIFT, MUX, ENCODE, DECODE |
| NETSYM | PWR, GND |
| PASS | XTAL, XFMR, TEST, RES, IND, FUSE, CAP, BAT |
| PLD | PLD, EPLD, PAL, FPLA |
| PROC | COPROC, MPERIPH, MCTRL, PROC, MEMCTRL, MEMMAP |
| REG | REG, SHREG |
| THYR | UJT, TRIAC, THYR, SCR, DIAC |
| TRANS | TMOS, PNP, NPN, JFET, MOSFET |
| XCVR | XCVR |

# Design Kits and Sheet Templates

## 13

This chapter is about how to customize DesignWorks to fit your particular design application by installing and creating design kits. Topics covered here include how to install and use the design kits supplied with DesignWorks, making simple changes to the design kits supplied (such as adding your company logo to the sheet templates), and creating your own design kits from scratch.

A design kit groups together all the files that are relevant to your particular design process, and may include:

Design templates
Example designs
Symbol libraries

## Design Templates vs. Sheet Templates

This chapter covers two related topics: *design templates* and *sheet templates*. Both of these things are standard DesignWorks design files from which initial setup data is imported, but they have different usages:

> A design template is intended to be used as the source for a New Design command. When a design template is selected in the New Design box, the design template file is opened and all the circuit data and settings it contains become the starting point for a new design. More information on this process is given in "Creating Design Templates" on page 342.
>
> A sheet template is intended for use with the Import Sheet Info command and is used to set only the sheet border, title block, print setup and other settings related to the printable sheet. The Import Sheet Info command is covered in "Importing Sheet Settings from Another Design or Page" on page 352.

There is no reason why a design template cannot be set up with all the sheet settings you need for your design. This eliminates the need to ever refer to a sheet template. However, sheet templates can be used to apply different border styles and settings to different parts of a single design. In addition, separating out the sheet functions can reduce the number of design templates you need to have available. The design template can be used to select the application type, attribute settings, script setup, etc. A sheet template can then be applied to customize the sheet border as a separate operation.

Netlist and report formats

Error checking scripts

Data entry boxes

Back annotation formats

Custom menus

Documentation.

Obviously, not all these items are required for every application. If your design process requires you to simply create a schematic for hardcopy purposes, then your design kit may contain only a design template and the symbol libraries you need. The benefit of a design kit is that it groups everything together under one heading and makes it easy to install or remove a complete set in one simple operation.

Although you are certainly free to make any of the above items without specifically creating a design kit, the simple extra step of using the standard design kit organization will make your design process clearer and assist others in your organization that may use the same resources.

DesignWorks is shipped with a variety of design kits for the various common applications. However, you will likely want to customize these to some degree, whether it be simply adding your company logo to the sheet templates, or creating an extensive set of libraries and scripts for an exotic design process.

## Using Design Kits

A design kit is a collection of items that affect almost all aspects of DesignWorks operation. For this reason, there is no single menu command or operation that is used to access a design kit. The creation and usage of libraries, scripts, templates and other design kit items are covered in their respective sections of this manual. The purpose of this section is to describe how design kits are installed and activated as a group.

DesignWorks has the concept of active vs. inactive design kits. An active design kit is one whose resources (libraries, templates, etc.) appear when the program is run. An inactive design kit may be installed on your disk and yet not appear when the program is started. This allows you to hide items that are not currently in use, thus simplifying program operation and reducing errors. When a design kit is needed for a specific project, it can be activated again in a single operation.

## Choosing and Activating a Design Kit

If one of the design kits supplied with DesignWorks exactly matches your application, then the choice will be very clear. For example, if your primary use for schematics is as a front end for the Douglas Professional Layout PCB design system, then the Douglas PCB design kit is the obvious choice. If there is no supplied kit that exactly matches your usage, or if you have a combination of requirements, then you will have to choose one or more of these approaches:

Choose multiple existing design kits and understand how the attribute and reporting requirements of each application can be met.

Choose an existing design kit that is similar to your application and modify it to meet your needs.

Create a new design kit from scratch.

Even if a supplied kit is exactly what you need, it is likely that you will want to change some aspects of its operation, such as placing your company logo on the design template or changing the bill of materials format. These minor issues are addressed in "Making Simple Design Kit Changes" on page 340.

If your primary application is PCB-related, but we don't directly support that package you are using, then chances are that one of the existing design kits will be close to what you need. In this case, the primary issue will be the netlist format required by the target system and its naming and pad library issues. If it is compatible with one of the common systems already supported, there may be little extra that you have to do.

***For more information on these issues, see "Before Starting a Major Design" on page 117.***

## Activating a Design Kit

To activate an already-installed design kit, that is, to make its resources available next time you start DesignWorks, follow these steps:

Select the DesignKit item in the Tools menu.

Select the desired item in the list of available design kits.

Click the Activate button.  This will move it to the list of active design kits.

Click the Done button.  This will update the DesignWorks Setup file with the necessary changes.

If you want to use the design kit now, quit and restart DesignWorks.

**NOTE: The newly-activated design kit will not be available until next time you start DesignWorks. Design kit items like design templates, libraries and custom menus are only located as the program starts up.**

If the design kit you want doesn't appear in the list of available items, there are a number of possible causes:

The design kit's files have not been installed in the correct directory. As installed, DesignWorks looks in the Design Kits directory, but this can be changed by making apprpriate entries in the DesignWorks Setup file.  See "Specifying Additional Setup Files - SETUPFOLDER" on page 389.

The design kit's file organization is not correct.  In order for the DesignKit tool to locate it, the design kit must be in a specific location.  See "Design Kit File Organization" on page 349 for more information on how these directories are organized.

**NOTE: A design kit is activated by placing a SETUPFILE keyword in the DesignWorks Setup file that points to the design kit's setup file. Since the DesignWorks Setup file is a text file, it is also possible to make these changes manually if you feel there is a need for a more complex file organization. See Appendix D—Setup File Format on page 387 for more information on the setup file format.**

## Installing a Design Kit

In general, a design kit is installed by placing its directory inside your Design Kits directory, however, the exact procedure will depend on the source of the design kit:

If you want to install a design kit that was provided with the DesignWorks release but not installed when you installed the program, then run the Installer program on the release disks and follow its instructions.

**WARNING: If you have modified the design kit directory structure by changing the standard directories specified in the DesignWorks Setup file, the installer will not be aware of this. See the comments in the ReadMe file on the installation disks.**

If you want to install a design kit that was provided separately, then following the instructions provided with it.

If you are creating a new design kit, then refer to “Design Kit File Organization” on page 349 for information on design kit directory organization.

**NOTE: Installing a design kit on your disk does not immediately make its resources available from within DesignWorks. You have to activate it using the DesignKit tool and then restart the program.**

## Deactivating a Design Kit

To deactivate an active design kit, i.e. to hide its resources so they don’t appear next time you start DesignWorks, follow these steps:

Select the DesignKit item in the Tools menu.

Select the desired item in the list of available design kits.

Click the Activate button. This will move it to the list of active design kits.

Click the Done button. This will update the DesignWorks Setup file with the necessary changes.

The next time you quit and restart DesignWorks the design kit will not appear.

**NOTE: The deactivating process simply removes the SETUPFILE line associated with the design kit from the DesignWorks Setup file. It does not remove any of the design kit’s files from your disk.**

## De-installing a Design Kit

To de-install a design kit (that is, to completely remove its files from your disk), follow these steps:

Deactivate the design kit using the steps outlined in "Deactivating a Design Kit" on page 339. If you remove the files from your disk without doing this, you will get an error message each time you start DesignWorks. This is due to the entry in the DesignWorks Setup file that points to the active design kit.

Using the Finder, locate and open the Design Kits folder inside the DesignWorks folder. Drag the entire folder containing the design kit in question to the Trash.

## Copying a Design Kit

All files related to a design kit a stored in the design kit's folder, which is normally found in the Design Kits folder inside "/Users/<user>/Library/Application Support/ DesignWorks" folder. To copy a design kit, for example, to form the basis for a new one you are creating, simply duplicate the folder containing the source design kit, then rename the folder of the copy to an appropriate new name.

You should also rename the library, script, template and other files in the copy to suit your new application. Although this is not a technical requirement of DesignWorks or the file system, there is no way to distinguish between two items with the same name in the New Design template list, library palette and other areas.

# Making Simple Design Kit Changes

This section covers some simple design kit changes that almost all users will want to make. For more detailed information on design kit structure and creating one from scratch, see "Creating a Design Kit" on page 341.

**IMPORTANT: Don't modify the design kits provided with the package without copying them first! You may wish to return to them later if your application changes. The copying procedure is described in "Copying a Design Kit" on page 340.**

## Modifying a Design Template

The templates that are provided in a design kit are simply normal design files that the attribute fields, hierarchy mode, sheet graphics and other settings appropriate for a specific application. When you do a New Design and select a template, the design is simply opened and then disassociated from the original file so that it cannot accidentally be saved back onto the template.

For this reason, creating a template is simply a matter of setting up a design exactly the way you want it except with no circuit elements in it, and saving in the Templates folder inside the design kit's folder. A custom title block can be created using the normal text and picture pasting operations described in "Creating a Title Block" on page 355.

***For more complete information on sheet size settings and other sheet issues, see "Setting Sheet Sizes and Borders" on page 350.***

## Modifying a Netlist or Bill or Materials Format

All text netlists and reports are generated by the Scripter tool. The format of a report is determined by a script file that contains commands to extract and modify data from your design. These reports can be customized to change the format or to add new text fields that may be specific to your application.

The scripts associated with a design kit are stored in its Scripts folder and can be created or edited with any standard text editor application.

***See the DesignWorks Script Language Reference (separate manual on disk) for more information on modifying scripts.***

# Creating a Design Kit

This section summarizes the items needed in a complete design kit and points you to other parts of the manual for detailed information. Obviously, some of these items may not be relevant in all situations. In most cases you can start with one of the existing kits and make the modifications that suit you.

A well-dressed design kit should have:

Design templates with predefined sheet size, title block, border, attribute fields, hierarchy mode and other settings.

Symbol libraries or file aliases pointing to common libraries.

Scripts for netlists, reports, error checking, design conversion, data entry, etc.

Example designs illustrating a simple design process.

A "setup" file which defines the custom menus for starting scripts and the location of the design kit items.

Last, but not least, a "ReadMe" text file describing the purpose, installation and usage of the design kit.

## Creating Design Templates

In DesignWorks 4.7, a design template is simply a normal design file that has its sheet size, attribute fields, hierarchy mode and other settings predefined for the application at hand. The simplest way to create a template may be to take an existing design that is set up the way you like it, delete all the circuit elements and extra pages out of it and save it in the appropriate template directory.

When you select the New Design command and select one of the template files listed, DesignWorks just reads the file in the normal way, then renames it "Design1" (or the next available number), and disassociates it from the original file so that it cannot accidentally be Saved on top of the template. In all other respects, New Design is the same as doing an Open on the template design. All the settings and contents of the design template file become part of the new design.

### Contents of a Design Template

The template should be set up so that the user can just start in placing device symbols and signal lines without worrying about whether the border will fit on the printer, the netlist will come out OK, etc. This table defines the items that you may want to include or set up in a template file, and gives you a reference to the section in the manual that discusses the issue.

| Design Kit Issue | Where to Look |
|---|---|
| A title block | “Creating a Title Block” on page 355 |
| A sheet background, or auto-border settings | “Setting Sheet Sizes and Borders” on page 350 |
| A printer page setup, possibly with the reduction set as needed. Note that there is a new “Scale to Fit” sheet option that allows automatic scaling to fit your printer | “Setting Sheet Sizes and Borders” on page 350 |
| All built-in attribute fields that are not needed should be hidden, i.e. the “In Primary List” box should be off in the Define Attribute Fields box. | “Defining a New Attribute Field” on page 192 |
| All attribute fields required for netlisting and other operations should be pre-defined. Ones that require user entry should be “primary”, others should not be. | “Defining a New Attribute Field” on page 192 |
| A design attribute field can be created to list the device attribute fields that will be used to specify power and ground pins on devices. | The entry "Reporting Power and Ground Nets" in *the DesignWorks Script Language Reference (separate manual on disk)*. |
| The hierarchy mode should be set as appropriate. This may affect the usage of attribute fields, and therefore which items you want to make visible to the user. | “Choosing a Hierarchy Mode” on page 238 |
| The DesignType design attribute field should be set to a unique text string so that the netlisting and other scripts in the design kit can check that they are operating on an appropriate design type. | “Creating Netlists and Reports” on page 346 |
| Optionally, the Script.Open design attribute can be set to the name of a script file to run each time the design is opened. This can prompt the user for a revision level, offer warnings, hints or general encouragement, or whatever. | |
| Optionally, some notes on usage of the design template in a miscellaneous text block on the sheet. These can be deleted by the user when no longer needed. | |

**NOTE: Some users like to have a different format for page 1 of a design, i.e. for a title or index page. It is quite feasible for the template design to have 2 pages with the first one set up differently. The existing Sheet Info mechanism can be used to allow the user to apply different sheet settings to different parts of the design later, if needed.**

### Naming a Design Template

Technically, the name of a template file can be any valid file name. However, we suggest following the following informal convention in order to minimIze user confusion. The name should contain the following items:

Unique keyword identifying application type, e.g. “PSpice”.

A letter or keyword identifying the sheet style, e.g. “ANSI-B”.

A letter or keyword identifying the hierarchy mode, e.g. “Flat”.

For design kits for use on Windows 3.1-based systems, names must be limited to 8 characters, which obviously limits creativity. In these cases, we suggest using single-letter codes for sheet style and hierarchy mode and separating them by hyphens, as in: “PSPCE-BF.CCT”.

### Working from an Existing Design Template

If you are modifying an existing template that is used in an application similar to yours, here are the minimum things that you need to change:

If you are also modifying the report and netlist scripts in the design kit, the DesignType design attribute field should be set to different value. This ensures that these scripts can recognize the design type and warn the user if it’s not appropriate. Note that you also need to modify all scripts that check this value to match.

If the template has text notes on the schematic, modify them to reflect the new usage of the template.

The template should be saved under a different name that reflects its new usage.

## Creating Symbol Libraries

The creation of symbol libraries is covered in detail in Chapter 12—Device Symbols on page 271. However, the following points should be noted on how libraries fit in with the design kit concept:

DesignWorks 4.7 supports Macintosh file “aliases”. This allows a file to appear in more than one place without duplicating the storage for the file. In particular, libraries can be shared between many design kits by placing the libraries themselves in a common directory and placing alias files in the design kit directory. Alternatively, you can simply make an entry in the design kit’s setup file to point to shared resources without duplicating the file or creating an alias.

The Scripter has the ability to map text values based on a table provided in the script. This allows you, for example, to use the standard 74xx libraries provided with DesignWorks but to map the package codes or part names to suit your PCB layout system. In many cases this will obviate the need to modify the libraries themselves. See the entries for $MAP and $CHARMAP in the DesignWorks Script Language Reference (separate manual on disk).

## Creating Scripts

The scripting language in DesignWorks has a variety of uses in generating netlists and reports, checking for errors and simplifying data entry procedures. Script usage and creation is covered in detail in the DesignWorks Script Language Reference (separate manual on disk), but this section outlines some issues that are relevant to the overall design kit structure.

### Note to Users of pre-4.0 DesignWorks Versions

In general, netlists and reports are generated in the same way they were in previous versions, although you may wish to note these points in converting to the new version:

> There should be a high degree of compatibility with existing report forms, although they should be tested carefully in the new version. Some keyword and command changes may affect a few report formats. See more information in the DesignWorks Script Language Reference (separate manual on disk).
>
> Many new report features have been added to enhance error checking and other capabilities. You may want to check existing report forms and consider implementing the following changes using new command language features:
>
> See if any parts of the report script can be merged. In older versions, the only way of performing different formatting on different types of devices was to use the $FIND $DEVICES command multiple times. In many cases the new conditional operations mean that many of these sections can be merged into one.
>
> Consider adding more error checking, or breaking out the error checking capabilities into a separate script. You can enforce or suggest that a user run the error check script before generating other reports.

## Creating Netlists and Reports

The Scripter tool can be used to generate netlists and reports in a wide variety of formats. In creating such reports, you migh want to consider incorporating these features that tie in with other elements of your design kit:

Check the "DesignType" attribute value in the design to make sure the report is being generated on an appropriate type of design.

Create a custom menu (in the design kit's Setup file) to fire up the script instead of having to locate the file each time.

Consider adding error checking, or breaking out the error checking capabilities into a separate script. You can enforce or suggest that a user run the error check script before generating other reports.

Consider the optional generation of a "transcript" file that will allow the user to trace any automatic assignments or error conditions that may occur.

## Error Checking

Many of the scripting features are intended to implement design error checks. You may wish to consider using the following script features in your error checking scripts:

Check for duplicate or invalid pin numbers.

Use regular expressions to check the general format of names or values.

Count items meeting certain criteria and generate totals.

Use value tables to check whether attributes are drawn from an allowable set.

Error checking scripts can made more interactive by developing them for use in conjunction with the ErrorScript tool. This allows the user to locate objects on the schematic one at a time and view error information in a more user-friendly manner. See a complete description of ErrorScript in Chapter 8—Tools for Searching, Browsing and Error Checking on page 155.

## Data Entry

Scripts can display data entry boxes in various formats to allow the user to enter arbitrary text data. This data can be assigned to design, device, signal or pin attributes or can be used to control the flow of script operation. Two data entry mechansisms are available:

> The Scripter itself allows simple warning alert boxes or simple text entry boxes to be displayed as part of script operation. The values generated by these inputs can be used just like any other text values.
>
> The Scripter can invoke the more general capabilities of the Prompter tool for cases where more sophisticated value lists and text entry capability is desired.

Applications of these features include:

> Prompting for global design data, e.g. the revision level, text parameters to pass to external tools, etc.
>
> Prompting for values for selected devices, e.g. PCB package codes, SPICE parameters, etc.

## Back Annotation

The Scripter has the ability to read line-oriented text files and extract values using the regular expression features. These values can then be used to locate objects and assign attribute values. This can be used to back-annotate attribute data from a variety of external systems. See the entry "Text File Input" in the DesignWorks Script Language Reference (separate manual on disk) for more information.

## Invoking Scripts

Scripts can be invoked directly by the user in a number of ways or automatically in response to other user actions:

> Selecting the Scripter tool in the Tools menu will prompt for a single script file to execute.
>
> A script can be directly executed from a custom menu item defined in a setup file. See “Adding a Custom Script Menu - SCRIPTMENU” on page 391.
>
> A number of design attribute fields can specify script files to execute under various circumstances: when a device is placed, when a design is opened, etc.
>
> A general capability to place a script in an attribute field to generate

the value for another field is now available.

The new ErrorScript tool will allow scripts to be executed and the results viewed in specific ways.

## Creating a Design Kit Setup File

Each design kit must have a “setup” file which specifies the location of its libraries, scripts, templates, etc. When a design kit is “active”, i.e. it has been activated using the DesignKit tool, the kit’s setup file will be read as DesignWorks starts up and the items specified in the file will be opened or acted upon.

A typical design kit setup file will look like this (this example is based on the Generic PCB design kit provided with the package):

```
LIBRARYFOLDER ::PCB Libraries:;

TEMPLATEFOLDER :Templates:;

EXAMPLEFOLDER :Design Examples:;

ERRORSCRIPTFOLDER :Error Scripts:;

SCRIPTFOLDER :Scripts:;

SCRIPTVAR _GPCB_Dir, :Scripts:;

SCRIPTMENU Generate Netlist, Generic PCB, :Scripts:GPCB Netlist Script;

SCRIPTMENU Error Check, Generic PCB, :Scripts:GPCB Error Script;

SCRIPTMENU Enter Package Code, Generic PCB, :Scripts:GPCB Pkg Script;
```

Most of the lines in this file should be fairly self-explanatory, but note these points:

All file or directory specifications are relative to the folder containing the setup file itself, i.e. usually the design kit’s folder. For this reason, most of these entries will be the same for all design kits. For example “:Templates:” refers to a folder called Templates inside the design kit’s folder.

The item “::PCB Libraries:” refers to a directory called PCB Libraries in the directory one up from the design kit’s directory. In other words, the same directory that the design kit’s directory is in, normally “Design Kits”. In this case, the purpose is to have a separate folder for libraries that are shared by all the PCB-related design kits.

The SCRIPTMENU lines add a menu command to the menu bar and associate a script with each command. This provides a direct way of executing a specific script at any time.

***For a more complete description of setup file commands, see Appendix D—Setup File Format on page 387.***

Here are some points to keep in mind when creating the setup file:

The setup file can be created using any word processor or text editor application that can produce plain text files. If using a word processor, ensure that you use a "text only" option when saving the file.

The setup file must be given a name that ends with the word "Setup" in order for it to be located by the DesignKit tool.

## Creating Example Designs

Every design kit should contain at least one design example that is known to generate correct netlist and report output and that illustrates the use of important attributes and features of the design kit.

## Creating a ReadMe File

Every design kit should include a ReadMe file describing:

The function of any custom menus installed by the kit.

The contents of any symbol libraries included or a list of the common libraries required.

Usage of all attribute fields relevant to this design kit.

How to install the design kit.

The value used in DesignType.

The ReadMe file should be created using a simple text editing application like SimpleText to ensure that all users can read it.

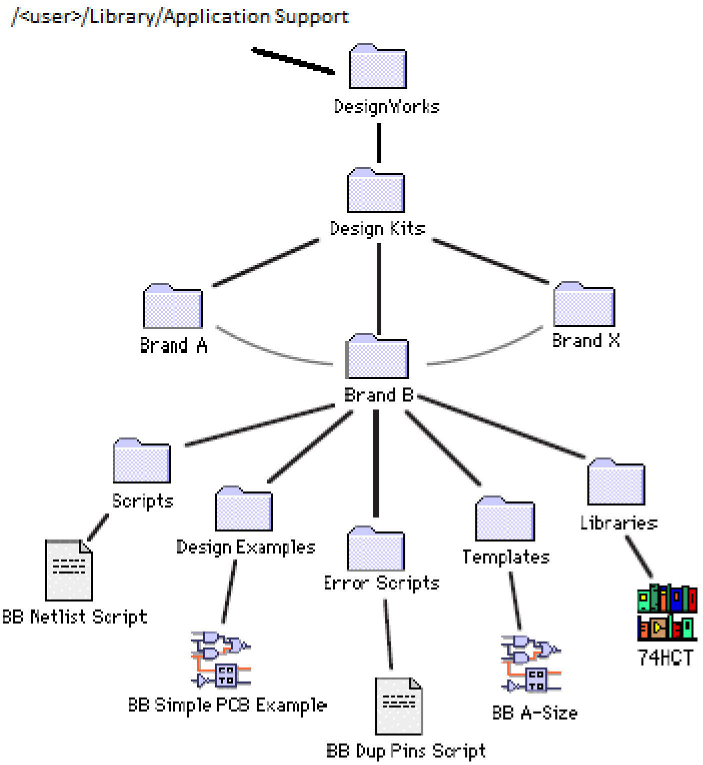
## Design Kit File Organization

Although you can technically install the elements of a design kit anywhere you like, a consistent organization ensures that:

All file references are relative to a single root, so that the whole installation is easily ported to another system without changes.

Users will understand where to find design kit items without extra explanation.

Here is the directory layout that is used for DesignWorks 4.7 installations:



In many cases, it may be desirable for design kits to share large files such as libraries. For example, most PCB design kits will be able to share the same symbols for common parts. These items can be put in a folder inside the Design Kits folder, i.e. in the same area as the design kits themselves. Design kits can refer to shared items either by specifying a path or making a file alias in the design kit's own folder.

## Setting Sheet Sizes and Borders

DesignWorks does not have any built-in knowledge of sheet sizes. Sheet border information can either be customized for each design or template or imported from an existing design file. This section describes the methods used for setting the sheet size and border for a schematic and the use of various options for adjusting the schematic border to the available output device.

DesignWorks supports very flexible use of plotters and printers for output on a variety of media. For this reason, there can be complex interactions between the drawing area allocated for each page and the way the schematic is presented when it is output on various kinds of output devices. Various settings allows you to:

> Have the schematic border adjust automatically to the current printer page setup.
>
> Have the schematic border retain a fixed logical size and have the printer scaling adjust automatically to fit the schematic on a single sheet.
>
> Have the schematic border retain a fixed physical size and be broken into the required number of printed sheets for output.

The following information is considered to be part of the complete sheet definition:

> The overall height and width of the drawing area.
>
> The origin and spacing of the location grid.
>
> The format of text used in the border.
>
> Whether or not the border size should be linked to the current printer page setup.
>
> Whether or not the page should auto-expand to multiple sheets.
>
> Any text and picture objects marked as "border" items.

DesignWorks allows sheet size and border layouts to be set by one of two methods:

> 1) Importing an existing layout using the Import Sheet Info

## Sheets vs. Pages

In this manual the term "page" is used to refer to a circuit page, which is the single, contiguous area that is viewed in one circuit window. The term "sheet" is used to refer to the physical sheet of paper that the diagram is printed or plotted on. Depending on the sheet size settings, a page may correspond to one or more sheets.

command. This is the simplest method of setting the sheet size for a new design or template, but it assumes you have an existing design that has the desired settings.

2) Creating a custom layout using the Custom Sheet Info command. This command, and the associated text and picture capabilities, allows all aspects of the sheet size, drawing grid, border text style, title blocks and graphics to be set up to meet your drawing standards.

## Importing Sheet Settings from Another Design or Page

If you have an existing design or template file that has the sheet size and border settings that you want, you can import those settings into the current design using the Import Sheet Info command. This command performs this sequence of operations:

Deletes all text and picture objects marked as “Border” items from the current page or design.

Copies all text and picture objects marked as "Border" items in the source design or page into the current design or page. The Border switch must be set individually for each border text or picture item using the Get Info command (or Command-clicking).

Copies the Custom Sheet Info settings (sheet width, grid settings, border text settings, etc.) from the source design or page to the current design or page.

Optionally copies the printer page setup from the source design to the current design.

***For more information on the “border” setting in text and picture objects, see “Background and Border Objects” on page 111.***

Note these important points regarding this operation:

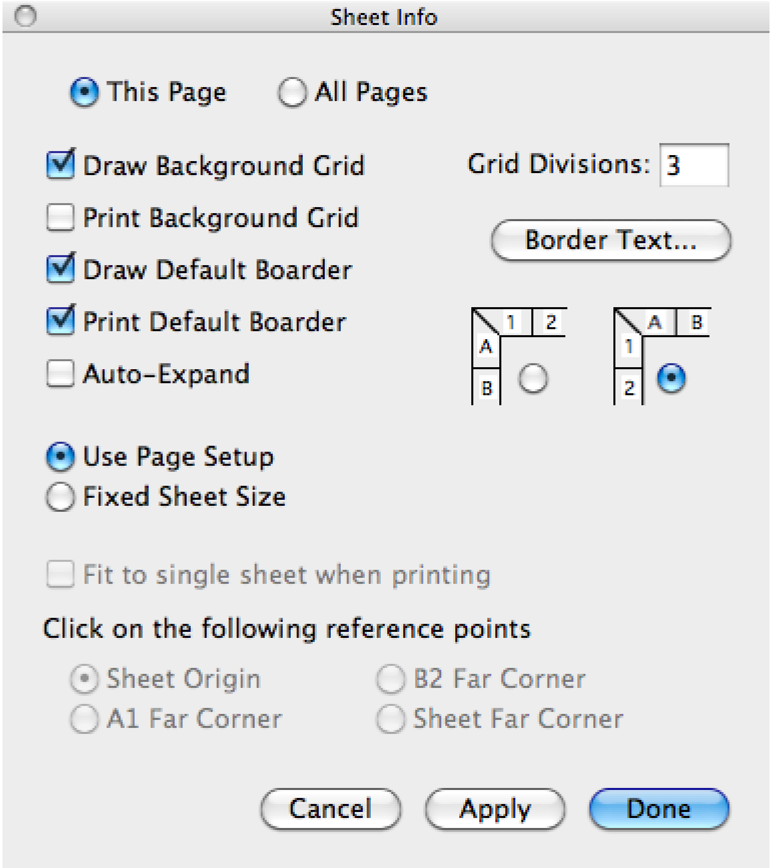
If the source for the import is a design file, only the items and settings on page 1 of the source design are imported. There is no provision for importing different settings from multiple pages.

The import operation does not affect any circuit items or attribute settings in the destination design.

## Summary of the Custom Sheet Info

## Settings

The Custom Sheet Info command displays the Sheet Info palette which allows custom sheet layouts to be created.



Sheet Info

● This Page ○ All Pages

☑ Draw Background Grid

☐ Print Background Grid

☑ Draw Default Boarder

☑ Print Default Boarder

☐ Auto-Expand

Grid Divisions: 3

● Use Page Setup

○ Fixed Sheet Size

☐ Fit to single sheet when printing

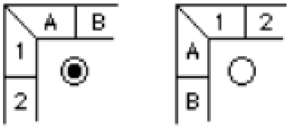
Click on the following reference points

● Sheet Origin ○ B2 Far Corner

○ A1 Far Corner ○ Sheet Far Corner

This table summarizes Sheet Info palette controls and options.

| | |
|---|---|
| **This Page/All Pages** | These buttons determine whether the changes made will be applied to the current page only or to all pages in the current circuit level when the Apply or Done buttons are clicked. |

| | |
|---|---|
| **Draw Background Grid** | This switch determines whether the gray background grid is drawn in the diagram on the screen. The setting for printed output is determined independently by the "Print Background Grid" item. |
| **Print Background Grid** | This switch determines whether the background grid is included in printed or PICT file output. |
| **Draw Default Border** | This switch determines whether the default sheet border is drawn around the drawing area on the screen. |
| **Print Default Border** | This switch determines whether the default sheet border is drawn in printed or PICT file output. |
| **Grid Divisions** | This number determines how many minor grid divisions are drawn per major division. The default value is 3, entering 1 in effect disables minor grid lines. This has no effect if both Print Background Grid and Draw Background Grid are off. |
| **Border Text** | This button displays the standard text style dialog box allowing you to select the text style for the default border. This has no effect if both Print Default Border and Draw Default Border are off. |
|  <br>**(Grid letter/number usage)** | These switches determine the usage of letters and numbers for sheet grid references. This setting will affect the position displayed on the tool palette, on Page Connectors, in the Pin List box and in report output. |
| **Auto-Expand** | If this switch is on, drawing will be allowed outside the drawing border and the drawing will be automatically expanded to multiples of the sheet size settings. NOTE: This of course means that the drawing will no longer fit on a single sheet and will have to be printed or plotted on multiple sheets. |
| **Use Page Setup** | Selecting this option causes the current printer page setup to be used to determine the sheet size. Any changes in page setup (using the Page Setup command in the File menu) will immediately affect the sheet border. |
| **Fixed Sheet Size** | Selecting this option causes fixed values to be used for the border size and position, regardless of printer page setup. |

| | |
|---|---|
| **Fit to Single Sheet When Printing** | Enabling this option instructs the program to automatically calculate a print scaling factor for each printed page so that the entire page fits on one printed sheet. This optionrelies on scaling functions being implemented in the printer driver and may not be available for all printers. |
| **Sheet Origin** | When this button is selected, clicking in the current schematic page will set the sheet origin point at that position. The sheet origin is the outer edge of the sheet in the corner containing the A1 grid position. The sheet origin can be in any of the 4 corners. |
| **A1 Far Corner** | When this button is selected, clicking in the current schematic page will set the far corner (i.e. farthest from the sheet origin) of the A1 grid rectangle. This allows the program to determine the size of the first grid space. |
| **B2 Far Corner** | When this button is selected, clicking in the current schematic page will set the far corner (i.e. farthest from the sheet origin) of the B2 grid rectangle. This is used to determine the size of all remaining grid spaces. |
| **Sheet Far Corner** | When this button is selected, clicking in the current schematic page will set the far corner of the sheet (i.e. opposite the sheet origin). |
| **Apply** | This button applies all settings currently displayed in the Sheet Info palette to the circuit. The palette remains displayed so further changes can be made. THIS CANNOT BE UNDONE!!! |
| **Done** | This button applies all settings currently displayed in the Sheet Info palette to the circuit and closes the box. These changes cannot be undone. |

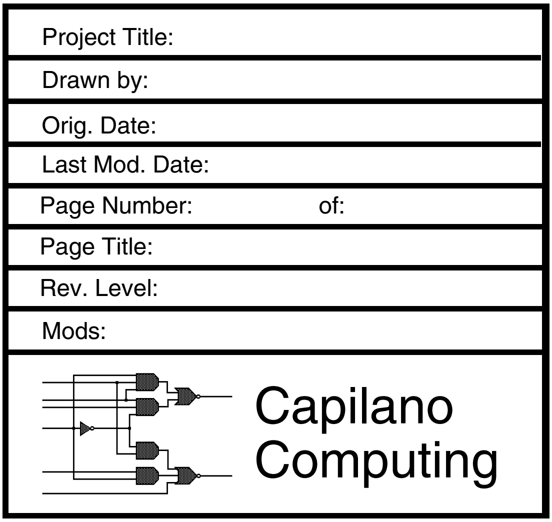
## Setting the Simplest Possible Sheet Layout

The sheet size options "Use Page Setup" and "Single Sheet" give the simplest possible sheet layout and are a good default for most design situations. With these settings, each circuit page will be printed on a single sheet of paper and any change in the printer Page Setup will be immediately reflected by a change in the drawing border.

## Creating a Title Block

DesignWorks has a number of features which can assist in creating title blocks which will contain information that is automatically updated. A simple title block can be created simply using the random text feature by clicking the pencil cursor in the desired location on the diagram. After the text has been typed, click on the text block to select it, then use the Get Info command to set the text size appropriately and create a frame.

A more striking title block can be created using a graphic item imported from a Macintosh drawing program, such as the following:



First, paste the graphic on the diagram using the Paste command then select it and set it to be a background and border object using the Get Info command. Once it has been converted to background, it will not be selected again and you can use text notations to draw on top of it.

**NOTE: Background objects can be selected by holding the Command and Option keys while clicking on them.**

**TIP: Text variables can be used to add dates and titles to the block. See "Using Text Variables" on page 110.**

### Creating a Fixed Sheet Size

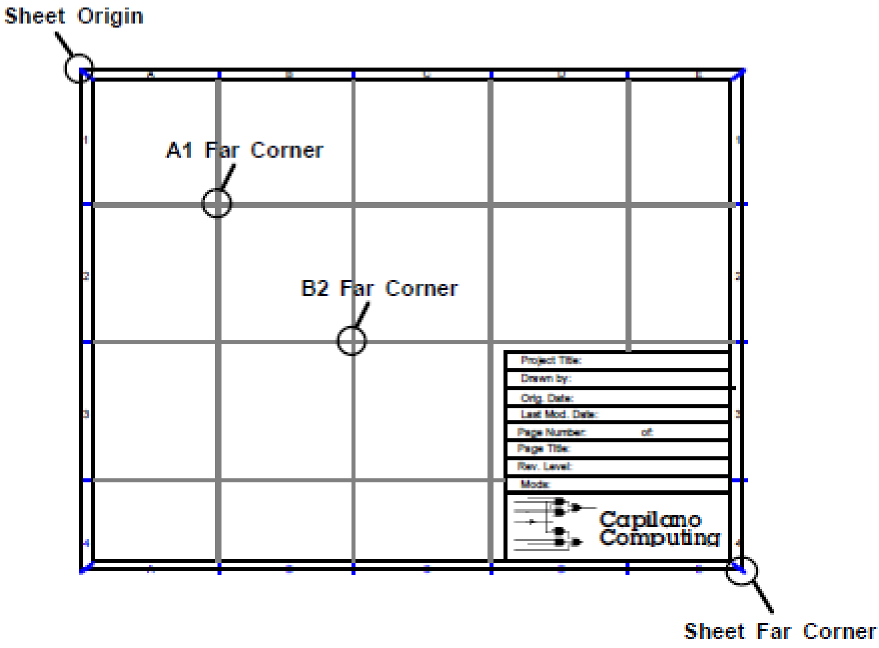
A fixed sheet size is one that does not vary with the current Page Setup printer settings. The fixed size that you determine can be considered a "logical" size since the actual output size is also affected by any scaling specified in the printer Page Setup. You have two options for determining how printer scaling is determined on output:

> If the "Fit to single sheet when printing" option in the Custom Sheet info box is OFF, then the current scale setting in the printer Page Setup will be used. If the schematic page will not fit on a single printer sheet, then it will be broken into multiple sheets as needed. The resulting image will be centered in the array of printed sheets.
>
> If the "Fit to single sheet when printing" option in the Custom Sheet Info box is ON, then the current scale setting in the printer Page Setup is ignored and a new scale is calculated for each page printed so that it exactly fits on one printed sheet.

In order to have a complete description of a sheet, you need to indicate to the program the following points:

> Point #1 - The sheet origin. This is the corner of the sheet where the A1 grid box is located. The origin point will be on the outside line of the border. Drawing will not be allowed outside this point.
>
> Point #2 - The opposite corner of the A1 grid rectangle, i.e. where it meets the B2 grid rectangle. This point is used to determine the start of the regular drawing grid.
>
> Point #3 - The opposite corner of the B2 grid rectangle. The space between Points 2 & 3 determines the grid spacing for the rest of the sheet.
>
> Point #4 - The far corner of the sheet. The point indicated will be the outside line of the border. Drawing will not be allowed outside this point.

In order to accurately click on these points in the circuit window, it is best to create a measured template picture using an external drafting application. This picture can be first pasted in the circuit at the desired location, then used as a reference for this procedure. Following is a sample A-size horizontal background picture with the reference points marked:



**NOTE: The sheet origin can be in any corner. It is the corner that will be the origin (i.e. the "A1" box) for the reference grid.**

Select the Custom Sheet Info command in the Drawing menu.

Click on the Fixed Sheet Size option. You will see the Sheet Origin button has become enabled. This indicates that you should scroll to the sheet origin position and click accurately on the origin point.

You will now notice that the A1 Far Corner button has been automatically selected. Click now at the opposite corner of the A1 grid rectangle.

Next, locate and click on the B2 far corner and sheet far corner points.

Select any other sheet option boxes desired, then click on the Done button. The sheet info palette will disappear and the displayed drawing area will be updated.

**TIP: If you are creating a very large template (e.g. ANSI E-size), follow these steps to create a suitable work area:**

**1) Open an existing sheet template file that is the same size or larger than the one you want to create. This ensures that you have a suitable size drawing area to work in.**
**2) Select the Custom Sheet Info command.**
**3) Select the Auto-expand and Use Page Setup options and click Done. This ensures that there is some white drawing area outside the sheet border in case your new template is larger than the original one.**
**4) Select the Reduce to Fit command in the drawing menu to see the entire work area.**
**5) If the sheet template you opened contains a border picture item, hold the Command and Option keys while clicking on it to select it, then use the Delete key to delete it.**
**6) Place your border pictures roughly centered in the drawing area and set the border corners and grid positions using the procedure given in the manual on pages 83 - 85.**

**TIP: Some border formats, like the ANSI standards, run right to the edge of the paper and no drawing is desired right around the outside border. This makes it more difficult to click on the precise locations of the sheet corners for the Custom Sheet Info command. In these cases, we recommend creating an "overlay" in your favourite drawing program that aligns with one corner of the border picture and has marks or lines in the desired positions for the sheet corners and grid lines. Paste this temporarily into DesignWorks on top of the border picture and use it to set the "Sheet Origin", "A1 Far Corner", "B2 Far Corner" and "Sheet Far Corner" positions. To check the grid alignment, turn on the "Draw Background Grid" option to make sure the grid lines appear where you expect. You can then delete the temporary overlay.**

## Creating Custom Sheet Border Graphics

Any picture or text objects can be made part of the sheet border. To do this:

select the object.

choose the Get Info command in the Options menu.

enable the "Make Background" and "Make Border" switches.

click OK.

The object will now be treated as part of the sheet background. To select it for deletion or editing, hold the Command and Option keys while clicking on it.

## Setting Text Styles

### Setting Pin Number Text Style

The text style for pin numbers is set globally for the entire design. It cannot be set individually for pins.

To set pin number text style:

1) Select the Design Preferences command in the Drawing menu.
2) Click on the Pin Text button.
3) Select the desired text font, size and style in the style box.
4) Click OK on the text box, then OK on the Design Preferences box.

Depending on the size of the design, there may be a short delay at this point while sizes and positions of text items are recalculated.

### Setting Attribute Text Style

The text style for attributes is set globally for the entire design. It cannot be set individually for each item or for each field.

To set attribute text style:

1) Select the Design Preferences command in the Drawing menu.
2) Click on the Attr Text button.
3) Select the desired text font, size and style in the style box.
4) Click OK on the text box, then OK on the Design Preferences box.

Depending on the size of the design, there may be a short delay at this point while sizes and positions of text items are recalculated.

### Setting Border Text Style

Text style can be set for the reference letters and numbers appearing in an auto-generated sheet border. To do this:

1) Select the Custom Sheet Info command in the Drawing menu.
2) Click on the Border Text button.
3) Select the desired text style.
4) Click OK on the text style box, then Done on the border info box.

### Setting Text Block Text Style

Text style can be set individually for each text block. To do this:

1) Select the text block by clicking on it. (If it is a background item, then hold the command and option keys while clicking on it.)
2) Select the Get Info command in the Options menu.
3) Click on the Text Style button.
4) Select the desired text style.
5) Click OK on the text style box, then OK on the text info box.

# Creating Multipage Templates

DesignWorks allows a circuit to be represented on multiple pages with logical connections between pages. When a new circuit is created, it is assumed to have a single page. A new page is added to the circuit by selecting the Pages command in the Drawing menu and clicking on the New Page button.

## Sheet Border Setup for Multi-Page Designs

When a new page is added to a circuit, all border information and text and picture objects marked as "border" items are copied automatically from the preceding page (i.e. what was formerly the last page). For this reason, it is most convenient to set up the first page of the circuit with the desired border arrangement before adding other pages.

Borders on all pages in a design can be updated later if needed using the Import Sheet Info command.

# Report Generation and Scripting

# 14

## Introduction to the Scripter Tool

This chapter provides basic information on using the Scripter tool for netlisting and error checking with existing scripts. Some information is provided on simple modifications to report generation scripts that can be made without an in-depth knowledge of the command language. Detailed information on the script language can be found in the DesignWorks Script Language Reference (separate manual on disk).

**NOTE: Many statements about the operation of netlist generation scripts must be considered to be generalizations. Since the script itself is essentially a program that gets executed, almost any aspect of its behaviour can be altered to suit the application. If you are using one of the design kits supplied with the package, check the ReadMe file provided in the design kit's directory for more information. This file will tell you if there are any special considerations in using the scripts provided.**

## General Information on Scripter

The Scripter tool provides powerful and customizable text report generation, error checking, back annotaton and design analysis capabilities. The tool works from a script defined in a text command language which allows you to read data from the design and from external text files, and to generate text data out to files and back into the design itself. DesignWorks includes a number of scripts for generating common industry netlist formats and performing error checking tasks. For cross-platform applications, the Scripter can directly generate files compatible with DOS, Windows or Unix systems.

The Scripter also works in conjunction with other DesignWorks tools, notably ErrorScript, to perform a variety of design checking and updating tasks.

***The ErrorScript tool is described in "Using the ErrorScript Tool" on page 166.***

### Output File Formats

Scripter creates text files which are intended to be incorporated into other documentation or transmitted to other systems. Files are saved with a file type of "TEXT", so they are accessible to almost any word processing program or text editor that runs on the Macintosh.

For creating human-readable printed reports, you will generally get the best results by converting the file to a font such as Monaco or Courier, which has fixed character spacing or by using tabs to space out columnar information. No font or formatting information is stored with the files.

### Output File Name

The default output file name is the design file name with " Report" added. However, the script file can set any desired file name and optionally prompt the user for a new name before the file is saved.

## Generating Standard Netlist and Report Formats

A netlist is generated by opening the design you wish to work from, then running the appropriate script. There are a number of DesignWorks commands that can cause scripts to be run, but the most common methods of generating a netlist are these:

If the design kit you are using has set up a custom menu, you can simply select the "Generate Netlist" or similar command in the design kit's menu.

**NOTE: The name of the menu and the items it contains are completely determined by the design kit's setup file.**

If you wish to run a script that is not associated with a custom menu item, follow these steps:

Select the Scripter command in the Tools menu. This will display a list of all scripts that are part of all active design kits plus any scripts that have been used since you started DesignWorks.

If the script you want is in the list, select it and click on Run Script, or simply double-click on the item in the list.

If the script you want is not in the list, click on the Add Script button. You will then be prompted to select a script file using a standard file open box.

**IMPORTANT: Many of the standard formats provided with DesignWorks rely on specific attribute data being entered in the design. Use the Format Notes button described below to get specific details on each script file, or check the ReadMe file provided with the design kit. If you use a report script that is not from the same design kit as the design template that you started with, make sure you know the requirements of the netlist script.**

***More information on invoking the Scripter is provided in "Scripter Operation" on page 368.***

## Common Changes to Standard Report Forms

It may be desirable to make minor modifications to one of the standard report scripts in order to tailor it to your design needs. This section covers some common changes that can be made without an in-depth understanding of the script file format. Each of these features is covered in more depth in the DesignWorks Script Language Reference (separate manual on disk).

**IMPORTANT: Before modifying any script that comes with DesignWorks, make a backup copy of the original so you will have a reference to go back to.**

Script files can be viewed and edited using a text utility such as SimpleText or any standard word processor. If using a word processor, be sure to save the file using a "Text Only" option so that no extra formatting information is included.

## General Rules

Report scripts consist of text with commands embedded in it. Items starting with a "$" are commands to the Scripter and items starting with "&" are references to attribute fields or scripter variables. Any other text encountered, including blanks, punctuation, and control characters, is simply passed through to the output file. This means that you can place almost anything in a script and it will become part of the output file. The only place where the Scripter is fussy about format is in the arguments to "$" commands, which usually have to meet some requirements. Anything appearing in braces "{}" is a comment.

## Default File Name

The default output file name for a report script is determined by the $CREATEREPORT command in a script file. Following is a typical example:

```
$CREATEREPORT($DESIGNNAME.NET) $PROMPT
```

The contents of the parentheses determines the default name of the output file. The keyword $DESIGNNAME substitutes the name of the current design file. All other characters inside the parentheses are used verbatim. The $PROMPT keyword indicates that a standard file box should be displayed to confirm the file name. If this keyword is not present, the file is saved in the same folder as the design without any prompt.

If no $CREATEREPORT command is present in the script (or if the script generates some output before it is encountered), a name will be generated by adding the word "Report" to the design name and prompting the user.

## Attribute Field Usage

Attribute field data is heavily used in generating netlist output files. Most of the standard script files are set up to use the pre-defined attribute fields. These are easily modified to use customized fields which you have added to your design.

Attribute fields are referenced in a script file using the form:

```
&fieldName
```

Wherever this format is used, any other field name can be substituted.

For example, a typical device listing will appear as follows in a script file:

```
$DEVICES$DEVNAME &Part
```

The "&Part" item will be substituted with the contents of the Part attribute field. Any other field name can be referenced as desired, or additional ones can be added to the line.

## Extracting Power and Ground Connections from Attributes

The Scripter tool provides two general methods of specifying an attribute field that is going to be used to list power and ground pins. The first, and most direct, method is the $SIGSOURCE command. You will frequently see a line like this one in a report script:

```
$SIGSOURCE(Ground)
```

This line indicates that any pins listed in a field named Ground on a device should be added to the net with the same name in the netlist. If you have power and ground fields that are commonly used in all your designs, you can simply add more lines like the one above to the report script. See the DesignWorks Script Language Reference (separate manual on disk) for more options on this keyword.

Another method which provides more flexiblity in specifying different power and ground fields for each design is the $DESIGNSIGSOURCE command, as in:

```
$DESIGNSIGSOURCE(SigSources)
```

This command specifies that a *design* attribute field named SigSources should be searched for a list of power and ground fields. This is in most ways equivalent to putting a list of $SIGSOURCE lines in a special report script for that design. See the entry for "$DESIGNSIGSOURCE" in the DesignWorks Script Language Reference (separate manual on disk) for more information.

***For a more complete discussion of the various methods for making power and ground connections, see "Power and Ground Connections" on page 223 and the DesignWorks Script Language Reference (separate manual on disk).***

## Scripter Operation

The Scripter can be invoked directly by a number of different DesignWorks commands and indirectly by using other tools that make use of Scripter functions. The following sections enumerate these various methods.

### Starting Scripter from the Tools Menu

Select the Scripter item in the Tools menu. The following box will

### Cross-Platform Compatibility of Scripts and Report Output Files

One of the well-known aggravations of the computer industry is the lack of agreement on line termination characters in text files across various computers and operating systems. In order to make it easy to work in a mixed operating environment, the Scripter provides a great deal of flexibility in reading input and generating output.

#### Line Termination in Script Files

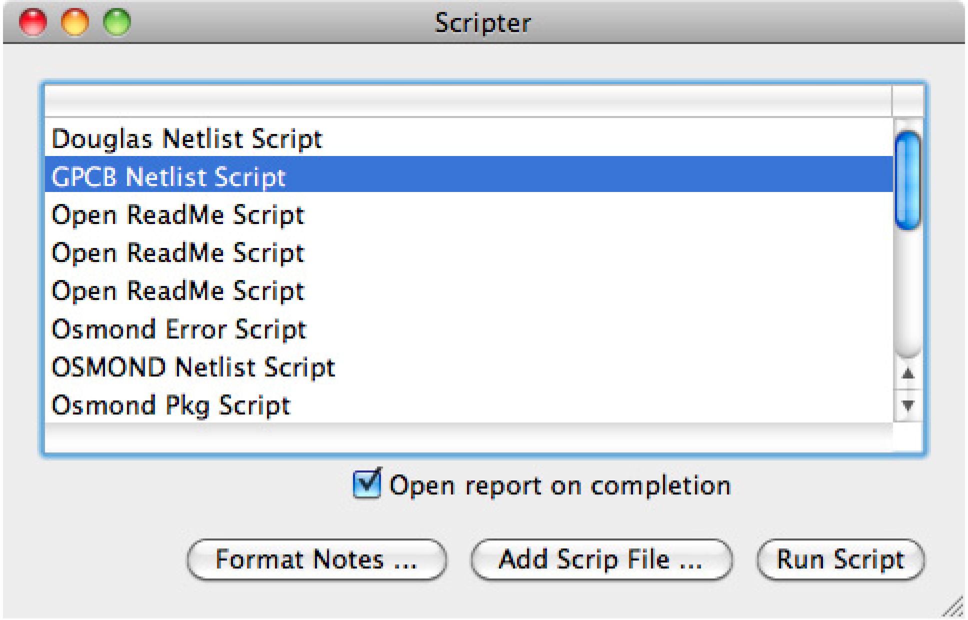
When reading a script input file, the Scripter accepts a single carriage return, a single line feed, or a combination of these in either order as a line terminator. This means that a text file from any Macintosh, Windows or Unix system can be used as input without affecting the output format. Script files created on a Macintosh can be used as input to the Scripter in DesignWorks for Windows and vice versa, even if they do not display correctly in a text editor on the executing platform.

#### Line Termination in Output Files

The line termination format that is used in the report output generated by the Scripter is determined by the $LINETERMINATOR command in the script file and does not depend on the line termination used in the script file. If no such command is present in the script file, then the output files will be terminated in the native format of the machine the script is being run on. Note that the Scripter also has the capability of generating a secondary "transcript" file for error logging, etc. This file always uses the line termination of the host computer system.

***See more information on line termination in the section "Line Terminators" in the DesignWorks Script Language Reference (separate***

appear:



☑ Open report on completion

By default, the last item used will be selected in the list. To run a script, simply select an item in the list and click the Run Script button, or simply double-click on an item in the list. The procedure after that is determined by the script file. Most script files will put up a standard file creation box to allow you to specify the name of the output file.

### Adding a Script to the Script List

The Add Script button allows you to choose a script file to add to the list of available scripts. Clicking this button will bring up a standard file box displaying all text files with names ending in "Script" or "Form". Select the desired script file by double-clicking on it.

**NOTE: Normally, the Add Script option only shows files with names ending in the word "Script" or "Form" to distinguish them from output files. You can select any text file, regardless of name, by clicking on the appropriate checkboxes in the file open box. "Script" is now the preferred ending for script files. "Form" is included for compatibility with earlier versions of DesignWorks.**

### Viewing Format Notes

The standard script files supplied will contain a number of comments explaining the format commands used. Some formats may rely on certain attribute fields being present in devices in the circuit. These notes can be viewed using the Format Notes button in the Scripter box:

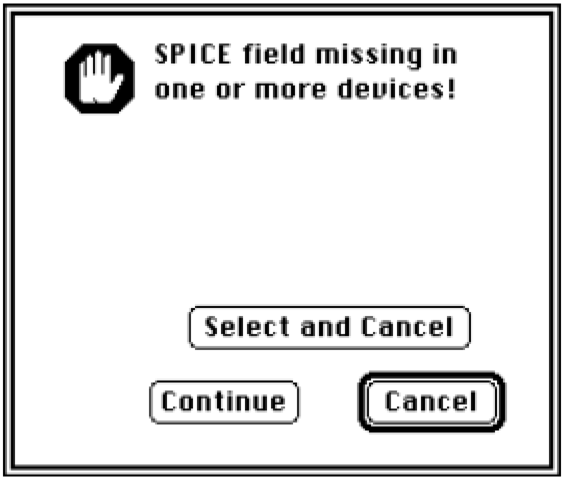
### Opening the Report on Completion

The "Open report on completion" option causes DesignWorks to ask the Macintosh system to open the file when the report is completed. If no other file type has been specified in the report script, this will normally cause the standard text utility "SimpleText" to be started up with the report output file as its document. It is possible for the script to specify other behaviour for this option.

***See the entry for the $SySTEMOPEN command in the DesignWorks Script Language Reference (separate manual on disk) for more information.***

## Circuit Checking

The report script file can contain optional commands to check for errors or missing data in the circuit before proceeding with generating the report. If such a check is encountered and a circuit error is detected, the following type of box will appear, although the text of the error message depends entirely on the script file:



SPICE field missing in one or more devices!

This box gives you the option of continuing despite the error, canceling the report, or selecting the offending devices or signals on the circuit diagram, then canceling. Note that the "Select and Cancel" option will only select objects on the current (topmost) circuit page, if there are multiple pages in the circuit.

## Device Reporting Options

The listing of devices and device internal circuits can be controlled using the Report Options button in the Get Info command for a device. The following options are available:

| | |
|---|---|
| **Omit from report** | Nothing about this device or hierarchical block will appear in the netlist. |
| **Report this device** | This device or block will appear in the netlist as if it has no internal circuit. |
| **Report sub-circuit** | The internal circuit of this device will appear in the netlist. I.e. in a hierarchical netlist, the internal circuit will be defined, in a flattened netlist, the internal circuit will be substituted and this device will not appear at all. |

***For more details on setting device information and options, see "Displaying and Setting Device Information" on page 85. For more information on setting the "report subcircuit" option while creating a hierarchical block symbol, refer to the Restrict attribute field in Appendix A—Predefined Attribute Fields on page 373.***

## Signal Reporting Options

The listing of signals can be controlled by setting the Omit from report option on a selected signal. For more information, see “Getting and Setting Signal Information” on page 98.

## Script Errors

Two general types of errors can occur in generating a report:

> Script syntax or execution errors - The Scripter command language is such that any items in the script file that are not recognized as commands are transmitted to the output file unmodified. Thus, most errors do not result in an explicit error message, but generate unexpected results in the output file. If an error is detected in an area of the script that requires a specific format, a message box is displayed showing the line number of the error. If such an error appears while running a script provided with DesignWorks, contact our Technical Support department for assistance. If you have created or modified the script yourself, refer to the DesignWorks Script Language Reference (separate manual on disk) for assistance in correcting the problem.
>
> Design data errors - A number of circuit data errors can occur during report generation, including unnumbered pins, excessive string length, missing or incorrect attribute values, etc. It is the responsibility of the script to check for the errors that may affect the type of output being generated. Errors can be displayed immediately in the form of an alert box, or may be logged to an output file, depending on the design of the script. See the Format Notes or ReadMe file supplied with the design kit you are using for more information.

# Appendix A—Attribute Fields

The following tables list fields that are predefined for each new design that is created. These fields cannot be removed because many of them are used internally by specific DesignWorks features.

***For more information on defining custom attribute fields see "Defining a New Attribute Field" on page 192.***

For convenience, the fields marked as "primary" by default are listed first since they are most commonly entered directly by the user. Thus, fields that are not used in a given application can be effectively hidden.

***For more information on setting the Primary/Secondary option, see "Primary vs. Secondary Fields" on page 174.***

The following information is provided in the table:

**Field Name**—The name of the field as it appears in the Define Attributes box.

**Used In**—The types of objects this field is used in.

Instance/Definition—Specifies whether the field data is kept with the definition or instance or a sub-circuit.

**Description**—How the field is used.

**NOTE: Many of the predefined fields are used for specific DesignWorks features. More details on the usage of these fields can be found in the manual sections covering those features..**

| **Field Name** | **Used In** | **Inst/Def** | **Description** |
|---|---|---|---|
| **AutoSym.Bottom** | Devices | Def | The string used in the "bottom pins" box in the Auto Create Symbol command in the DevEditor. See "Automatically Creating Symbols" on page 313. |
| **AutoSym.Left** | Devices | Def | The string used in the "left pins" box in the Auto Create Symbol command in the DevEditor. See "Automatically Creating Symbols" on page 313. |

| | | | |
|---|---|---|---|
| **AutoSym.Right** | Devices | Def | The string used in the "right pins" box in the Auto Create Symbol command in the DevEditor. See “Automatically Creating Symbols” on page 313. |
| **AutoSym.Top** | Devices | Def | The string used in the "top pins" box in the Auto Create Symbol command in the DevEditor. See “Automatically Creating Symbols” on page 313. |
| **BusInfo** | Pins | Def | Used only on bus breakouts to notate the names of the attached signals. Normally only updated using the Bus Pin Info command. See “Displaying a Bus Pin Annotation” on page 219. |
| **Category** | Devices | Def | Contains a terse device category code in the DesignWorks libraries. Can be used as an alternate device prefix by entering the name of this field in PrefixField. Not used internally. |
| **CctName** | Design | Def | Design file name. Sets the window title and name of next saved file. Normally only set by the Open and Save commands. |
| **CctOS** | Design | Def | The system (Mac or Windows) that was used to create the design file. This is used only for creating external circuit references for sub-circuit parts. Changing it will not affect the current design file. Normally only set by the Open and Save commands. |
| **CctPath** | Design | Def | The directory path name for the design file. This is used only for creating external circuit references for sub-circuit parts. Changing it will not affect the current design file. Normally only set by the Open and Save commands. |
| **DateStamp.Cct** | Design | Def | This field is set by the program to the current date and time each time the circuit is modified. This is intended to allow scripts to determine if a design has been modified since the last time the script was run. |
| **DateStamp.Dev** | Devices | Def | This field is set by the program to the current date and time when a device is placed in the circuit and not modified thereafter. This is intended for back annotation processes. See “Device Date Stamping” on page 153. |
| **DateStamp.Last** | Design | Def | This field is used by the program as part of the date stamping process. See “Device Date Stamping” on page 153. |

| | | | |
|---|---|---|---|
| **DateStamp.OS** | Design | Def | This is set to the system that the date stamping was performed on, to ensure cross-platform compatibility of dates. See “Device Date Stamping” on page 153. |
| **Delay.Dev** | Devices | Def | Used with the DesignWorks Simulator option. |
| **Delay.Dev.Max** | Devices | Def | Used with the DesignWorks Simulator option. |
| **Delay.Dev.Min** | Devices | Def | Used with the DesignWorks Simulator option. |
| **Delay.Dev.Typ** | Devices | Def | Used with the DesignWorks Simulator option. |
| **Delay.Pin** | Pin | Inst | Used with the DesignWorks Simulator option. |
| **Delay.Pin.Max** | Pin | Def | Used with the DesignWorks Simulator option. |
| **Delay.Pin.Min** | Pin | Def | Used with the DesignWorks Simulator option. |
| **Delay.Pin.Typ** | Pin | Def | Used with the DesignWorks Simulator option. |
| **Depth** | Devices | Def | Indicates nesting depth of device sub-circuit. Only used temporarily by the Report tool while generating hierarchical reports. This value is not maintained during editing operations. See the entry Depth Ordering in Pure Netlists in the DesignWorks Script Language Reference (separate manual on disk). |
| **Description** | Devices | Def | A short description of the device type. This is included in all standard DesignWorks libraries. Not used internally. |
| **Designer** | Design | Def | Intended to be used to display the designer’s name on the sheet. Not used internally. See “Creating a Title Block” on page 355. |
| **DesignType** | Design | Def | A value used to distinguish which design template was used to create the design. This is used to check that the appropriate netlist scripts are being used, etc. See “Contents of a Design Template” on page 342. |
| **DevPrefix** | Design | Def | Default prefix for auto-assigned device names. Normally set using the Device Naming and Packaging Options command. “Enabling Naming and Packaging Options” on page 127. |
| **ExtCctDate** | Devices | Def | Modified date of external design file containing device internal circuit. Normally set by the DevEditor. See “Working with External Subcircuits” on page 258. |

| | | | |
|---|---|---|---|
| **ExtCctLib** | Devices | Def | Name of library to search for an internal circuit. See See "Working with External Subcircuits" on page 258. |
| **ExtCctName** | Devices | Def | Name of external design file containing device internal circuit. Normally set by the DevEditor. See "Working with External Subcircuits" on page 258. |
| **ExtCctOS** | Devices | Def | Source system type for design file containing device internal circuit. Normally set by the DevEditor. See "Working with External Subcircuits" on page 258. |
| **ExtCctPath** | Devices | Def | Directory path for the design file containing the device internal circuit. Normally set by the DevEditor. See "Working with External Subcircuits" on page 258. |
| **Function** | Devices | Def | Contains a terse device function code. Can be used as an alternate device prefix by entering the name of this field in PrefixField. Not used internally. |
| **Ground** | Devices | Def | List of pins to be hooked to ground net in netlist, separated by commas. See "Power and Ground Connections in Attributes" on page 225. |
| **HierNameSep** | Design | Def | Contains the separator string used in generating hierarchical device and signal names for Report output. Normally set only by commands in a report script file. See the entry for the $HIERNAMESEPARATOR command in the DesignWorks Script Language Reference (separate manual on disk). |
| **InstName** | Devices, signals | Inst | Name applied to physical instance. This is normally used for the package name in hierarchical designs. Normally not used in Flat or Pure mode designs. See "Name vs. InstName" on page 141. |
| **Initial.Dev** | Devices | Inst | Used with the DesignWorks Simulator option. |
| **Initial.Pin** | Pin | Inst | Used with the DesignWorks Simulator option. |
| **Initial.Sig** | Signals | Inst | Used with the DesignWorks Simulator option. |
| **Invert.Pin** | Pin | Def | Used with the DesignWorks Simulator option. |

| | | | |
|---|---|---|---|
| **LibDate** | Devices | Def | "Last Modified Date" of the library the device was read or updated from, stored in encoded form. This is used by the Update from Lib command. See “Updating a Symbol from a Library” on page 278. |
| **LibName** | Devices | Def | Name of the library the device was read or updated from. This is used by the Update from Lib command. See “Updating a Symbol from a Library” on page 278. |
| **LibPath** | Devices | Def | Directory path of the library the device was read or updated from. This is used by the Update from Lib command. See “Updating a Symbol from a Library” on page 278. |
| **LibOS** | Devices | Def | Source system of the library the device was read or updated from. This is used by the Update from Lib command. See “Updating a Symbol from a Library” on page 278. |
| **LibType** | Devices | Def | The original name of the part when it was read from a library. This is used by the Update from Lib command. See “Updating a Symbol from a Library” on page 278. |
| **Name** | Devices,<br>signals | Def | Definition name, i.e. reference designator. This is the field set using the text tool on the schematic. See “Name vs. InstName” on page 141. |
| **Name.Prefix** | Devices | Def | Prefix used to create default Name value. Used by Packager and Auto-name. See “Setting the Name Prefix for a Symbol” on page 137. |
| **Name.Pt** | Devices | Def | Default position of Name attribute relative to device. See “Using Default Position Fields” on page 190. |
| **Name.Spice** | Devices | Def | Alternate prefix for SPICE-based simulators. See “Selecting an Alternate Prefix Field” on page 138. |
| **OKErrors** | Devices,<br>Signals,<br>Pins | Def | Used by ErrorFind tool to mark errors set as OK by user. Should not be set manually. See the entry Implementing Mark as OK in Error Checking Scripts in the DesignWorks Script Language Reference (separate manual on disk). |
| **Package** | Devices | Def | Device package code. This is included in all standard DesignWorks libraries. Not used internally. See “Selecting the Part and Package Type” on page 91. |

| | | | |
|---|---|---|---|
| **Package.List** | Devices | Def | List of package types that apply to this part type. See “Using Value List Fields” on page 188. |
| **PageRef** | Devices | Def | Used in Page Connector pseudo-devices only. Contains the auto-generated page references. See “Automatic Display of Page References” on page 219. |
| **PageRefFormat** | Design | Def | Format string for page references. Normally modified only using the Design Preferences command. See “Automatic Display of Page References” on page 219. |
| **PageRefWidth** | Design | Def | Max. number of items per line in page references. Normally modified only using the Design Preferences command. See “Automatic Display of Page References” on page 219. |
| **Part** | Devices | Def | Part type code. Used by Packager to assign gates to packages and in external netlists and bills of materials to identify the part type. See “Selecting the Part and Package Type” on page 91. |
| **Part.List** | Devices | Def | List of part types possible for this symbol. See “Using Value List Fields” on page 188. |
| **Part.Pt** | Devices | Def | Default position of Part attribute relative to device. See “Using Default Position Fields” on page 190. |
| **Permutable** | Devices | Def | This field contains a list of the swappable pins and gate sections in this part type. Parentheses "()" enclose items that can be swapped, while brackets "[]" enclose items that cannot be swapped. This is intended for use by external PCB layout packages. Not used internally. |
| **PinSequence** | Devices | Def | This field is used to determine pin order in netlist output formats that require specific device pin ordering, e.g. SPICE. This is <u>not defined</u> in the standard DesignWorks libraries. Not used internally. See the entry for the $DEVPINSEQUENCE command in the DesignWorks Script Language Reference (separate manual on disk). |
| **PkgLevel** | Devices | Def | An integer 0 to 3 specifying packaging action:<br>0 = normal<br>1 = lock and check<br>2 = lock and don't check<br>3 = ignore<br>Normally set using the Device Info or Get Info commands. See “Setting Device Packaging Options” on page 139. |

| | | | |
|---|---|---|---|
| **PkgPrefix** | Design | Def | Default prefix used by Packager to create a package name. Normally set using the Device Naming and Packaging Options command. See “Setting Design Attribute Fields Used By the Packager” on page 150. |
| **Power** | Devices | Def | List of pins to be hooked to power net in netlist, separated by commas. See “Power and Ground Connections in Attributes” on page 225. |
| **PrefixField** | Design | Def | Contains the name of the field to use as a name prefix for device packaging and device auto-naming. See “Setting Design Attribute Fields Used By the Packager” on page 150. |
| **Restrict** | Devices | Def | An integer value from 0 to 7 used to set access restrictions for the device's internal circuit. The value is the sum of the following:<br>1 = Don't report<br>2 = Don't package<br>4 = Don't Push Into<br>Normally set using the Get Info or Device Info command. See “Locking and Unlocking Subcircuits” on page 242. |
| **Revision** | Design | Def | Intended to be used to display the designer’s name on the sheet. Not used internally. See “Creating a Title Block” on page 355. |
| **Script.Dev** | Design | Def | The name of a script to run each time a device is placed. |
| **Script.Open** | Design | Def | The name of a script to run each time the design file is opened. |
| **Script.Pin** | Design | Def | Reserved |
| **Script.Sig** | Design | Def | Reserved |
| **SigPrefix** | Design | Def | Prefix used to generate default signal names. Normally set using the Signal Naming Options command. “Using Signal Auto-Naming” on page 229. |
| **Sim.InputMap** | Design | Def | Used with the DesignWorks Simulator option. |
| **Spice** | Devices,<br>Design | Def | Holds simulation parameters for SPICE-based simulators. Not used internally. See “Using DesignWorks with SPICE-based Simulators” on page 50. |
| **TestVectors.Cct** | Devices | Def | Used with the DesignWorks Simulator option. |
| **TestVectors.Dev** | Devices | Def | Used with the DesignWorks Simulator option. |

| | | | |
|---|---|---|---|
| **Unit** | Devices | Inst | Gate unit, e.g. a, b, c, etc. Set and checked by Packager or can be set manually. See “Attribute Fields Set By the Packager” on page 143. |
| **Unit.All** | Devices | Def | List of all units available in this part type. Used by Packager. See “Required Packaging Attributes” on page 143. |
| **Unit.List** | Devices | Def | List of all units in this part type having this symbol. Used by Packager. See “Required Packaging Attributes” on page 143. |
| **UnusedPins** | Devices | Def | List of unused pins in this part type. Not used internally. |
| **Value** | Devices | Def | Component value to appear in netlists. Not used internally. See “Selecting the Part and Package Type” on page 91. |
| **VisPin.List** | Pins | Def | List of pin numbers for this pin corresponding to the gate units in "Unit.List". Used by Packager. See “Required Packaging Attributes” on page 143. |

# Appendix B—Primitives

Every device on a DesignWorks schematic has a characteristic known as its *primitive type*. The primitive type is set when the part entry in the library is created and cannot be changed for individual devices on the schematic.

Primitive types fall into three general groups:

Schematic symbols: The two primitive types SUBCIRCUIT and SYMBOL fall into this category and are the normal primitive types used for creating schematic symbols. SUBCIRCUIT is the default type for symbols created using the DevEditor. There are no restrictions on the ordering or type of pins on these symbols.

Pseudo-device types: These are the symbols used for page connectors, power and ground symbols, etc.

**IMPORTANT: DesignWorks has very specific requirements for the order and type of pins on pseudo-devices. Refer to the following table for information. These rules ARE NOT CHECKED by the DevEditor.**

Simulation types: The majority of the primitive types defined in the following table are simulation primitives and are intended for use with the DesignWorks Digital Simulator option.

**IMPORTANT: The simulation primitive types should not be used for user-created symbols without a clear understanding of their function. See the Digital Simulator manual for more information.**

## Schematic Symbol Primitive Types

| Primitive Type | Pin Requirements | Comments |
|---|---|---|
| SUBCIRCUIT | No restrictions | Symbol having an optional internal circuit. This is the default for symbols created using DevEditor. |
| SYMBOL | ENo restrictions | Symbol with no internal circuit. |

## Pseudo-Device Primitive Types

**IMPORTANT: The pin requirements listed in the following table must be followed when creating pseudo-device symbols. These rules ARE NOT CHECKED by the DevEditor.**

| Primitive Type | Pin Requirements | Comments |
|---|---|---|
| BREAKOUT | Pin 1 is Bus Pin, followed by N Normal Pins, set to Input | Splits signals out of or into a bus. |
| SIGNAL CONNECTOR | Exactly 1 normal pin, normally set to Input | Used for power and ground connections. |
| PORT CONNECTOR | Signals - exactly 1 pin, Busses- exactly 1 bus pin with any number of internal pins | Makes a connection between the signal it is connected to and a like named pin on the parent device. |
| PAGE CONNECTOR | Signals - exactly 1 pin, Busses- exactly 1 bus pin and no internal pins | Exports the signal or bus globally across all pages at one level of a schematic. |

## Simulation Primitive Types

**WARNING: Do not use these without referring first to the Digital Simulation manual for more information!**

| Primitive Type | Pin Requirements | Comments |
|---|---|---|
| NOT | 1 Input, 1 Output, ... | Output complement of Input. Same for each gate. |
| AND | N Inputs, 1 Output | Output <- AND of all Inputs |
| OR | N Inputs, 1 Output | Output <- OR of all Inputs |
| XOR | N Inputs, 1 Output | Output <- XOR of all Input |
| XNOR | N Inputs, 1 Output | Output <- XNOR of all Input |
| D-FF | /Set, Data, Clock, /Reset, Q, /Q | |
| JK-FF | /Set, J, /Clock, K, /Reset, Q, /Q | |
| D-FF (non inv) | Set, Data, Clock, Reset, Q, /Q | Set and Reset are active high. |
| JK-FF (non inv) | Set, J, /Clock, K, Reset, Q, /Q | Set and Reset are active high. |
| BUFFER | 1 Input, 1 Output, ... | Output is Input. Same for each gate. |
| MULTI-PLEXER | N Data, M Select, /E, Output | Routes 1 of N inputs to the output. M is the smallest value such that 2M Š N |
| DECODER | N Select, M Output, /E, | Selects 1 of M outputs. N is the smallest value such that 2N Š M |
| REGISTER | N Inputs, N Outputs, Clock, Clear | Inputs are latched when clock is high, clear is async. |
| SHIFT REG | N Inputs, N Outputs, Clock, Shift/~Load, Shift-In | Shifts or loads data on clock. |
| ADDER | N A-Inputs, N B-Inputs, N Outputs, Carry In, Carry Out | Adds A and B w/Carry |
| COUNTER | N Inputs, N Outputs, Clock, Count/~Load | Counts up or loads on clock. |
| X-GATE | 2 Bidirectional, Control, ... | The 2 bidirectional pins are connected if control is active. Repeated for each gate. |
| RESISTOR | 2 Bidirectional, ... | Converts high drive to low drive for each resistor. |
| PULL_UP | 1 Bidirectional, ... | Outputs a high with a low drive level for each pin. |
| ONE SHOT | Clock, /Clear, Q, /Q | Retriggerable. Delay is 1 by default. Delay setting is used as pulse width. |

| | | |
|---|---|---|
| CLOCK | 1 Output | The delay setting is used as the high and low period of the clock. |
| PROBE | 1 Input | Displays Boolean input value on the schematic |
| SPST-SWITCH | 2 Bidirectional Pins | Makes a connection between the two pins when switch is closed, otherwise both pins are undriven. |
| SPDT-SWITCH | 3 Bidirectional Pins | Has pins A, B, and C; A and C are connected or B and C are connected. |
| HEX KEYPAD | Q0..3 Outputs, Strobe Output | Outputs a 4 bit hex value, strobe goes high when key pressed. |
| PUSH BUTTON | 3 Bidirectional Pins | A, B, and C Pins; A and C are connected or B and C are connected. Switch is momentary. |
| SETUP & HOLD | Clock, 1 Data (Input), Error (Output) | Error = X if Data changed after Setup or before Hold time, Error = Z otherwise. |
| GLITCH DETECTOR | Sense (Input), Error (Output) | Error = X if sense changes faster then delay setting, Error = Z otherwise. |
| UNKNOWN DETECTOR | Sense, Output | If Sense = 0 or 1 then Output = 0, Output = 1 otherwise. |
| SIMULATOR STOP | 1 Input | Performs requested task (stop simulator, generate report, ...) on rising edge. |
| LOGICBOX | | Represents LogicBox hardware interface. See LogicBox Manual. |

# Appendix C—Pin Types

Every device pin has a characteristic known as its *pin type*. The pin type is set when the part entry in the library is created and cannot be changed for individual device pins on the schematic.

***See "Setting the Pin Type" on page 307 for information on how to set the pin type while creating a device symbol.***

## What Pin Types are Used For

For many general schematic editing purposes, the pin type will be unimportant an can be ignored. However, pin type settings are important in the following cases:

Pin type settings are crucial if you plan to use the DesignWorks Digital Simulation option. The pin type of each pin determines the direction of signal flow and must be appropriately set for each symbol.

Pin type information is required in many netlist file formats for FPGA layout and digital simulation.

Correct pin type settings allow the ErrorFind module to check for fanout and multiple drive situations.

Other future analysis tools may use this information for timing and loading analysis.

## Pin Types Table

The following table lists the function of each of the pin types available in DesignWorks.

| **Pin Type** | **Description** |
|---|---|
| IN | Input - this is the default for pins created using DevEditor. This setting is used for all pins on discretes except those with some digital function. |
| OUT | Output - always enabled. |
| 3STATE | Output - can be disabled (i.e. high-Z) |
| BIDIR | Bidirectional |
| OC | Open collector output - i.e. pulls down but not up |
| BUS | Bus pin - This does not represent a physical signal but is a graphical representation of a group of internal pins, each having its own type. |
| LOW | Output - always driving low |
| HIGH | Output - always driving high |
| LTCHIN | Input to a transparent latch - this is used for calculating cumulative setup and hold times. |
| LTCHOUT | Output from a transparent latch - this is used for calculating cumulative setup and hold times. |
| CLKIN | Input to an edge-triggered latch - this is used for calculating cumulative setup and hold times. |
| CLKOUT | Output from an edge-triggered latch - this is used for calculating cumulative setup and hold times. |
| CLOCK | Clock input - this is used for calculating cumulative setup and hold times. |
| OE | Open emitter output - i.e. can pull up but not down. |
| NC | A no-connect pin |

# Appendix D—Setup File Format

A setup file is a text file which specifies some initial actions DesignWorks will take each time it is started up. The setup file specifies the following information:

Where to locate design kits.

Which libraries to open when the program is started.

Where to locate program "tools".

What custom menus to install.

Which colors to use in displaying objects.

Other options.

A setup file can be created or edited using your favorite programming editor or word processor. If you use a word processor, be sure to save the file using a "text only" option.

## Setup File Organization

DesignWorks expects to find a primary setup file when it starts up and uses the contents of this file to locate its design kits. This setup file must be called "DesignWorks Setup" and must be located in "/Users/<user>/Library/Application Support". Where "<user>" is the users home directory.

The standard DesignWorks license agreement does not allow multiple users to share the program over a network. Network licenses are available for this purpose.

The primary setup file will contain SETUPFOLDER commands that point to the active design kits, each of which will have its own setup file.

***See Chapter 13—"Design Kits and Sheet Templates" on page 335 for more information on design kit organization.***

## Setup File Format

A typical setup file looks like the following:

```
LIBRARYFOLDER :Basic Libs:;
TOOLFOLDER :Tools:;
```

The first word on each statement is a keyword which specifies a setup option. Statements are terminated by a semi-colon and can contain embedded comments in braces, as in `{ Comment }`. The remainder of this appendix describes the form and usage of the setup file keywords.

## Program Organization Keywords

The keywords in this section affect the way the program finds its basic components when it starts up.

### Specifying Folder Names in the Setup File

A number of the setup file keywords are used to specify folder locations for various purposes. Folders can be specified using either an *absolute* (i.e. starting with a disk name) or *relative* path. If the path begins with a colon, the first named folder is assumed to be in the same folder as the setup file containing the statement, otherwise the name is taken to be the name of a disk. The following are valid statements:

```
FOLDER :Libs:;
```

specifies a folder called Libs contained in the same folder that contains the setup file.

```
FOLDER LibDisk:4000 Series:;
```

specifies a folder called "4000 Series" on a disk called LibDisk.

```
FOLDER :Libs:7400 Counters:;
```

specifies a folder called "7400 Counters" which is contained inside a folder called Libs which is in the same folder as the setup file.

**WARNING: Inappropriate settings in any of these setup file commands may render the program unusable, since they affect where the basic program tools and design kits are located.**

## Specifying Additional Setup Files - SETUPFOLDER

```
SETUPFOLDER folderName;
```

The SETUPFOLDER keyword is used to indicate the location of additional setup files to be read when the program is started. This is intended primarily to specify the location of design kits to be activated when the program starts. Each design kit will have its own setup file that in turn indicates where its libraries, templates, scripts, etc. are located.

***More information on design kit organization is provided in "Design Kit File Organization" on page 349.***

## Locating Templates and Examples for the New Design Command

```
TEMPLATEFOLDER folderName
EXAMPLEFOLDER folderName
```

These two items specify the locations of the templates and examples, respectively, that are used in presenting the New Design dialog box. These will normally be present in a design kit's setup file to indicate the location of the templates and examples for that design kit. When a New Design command is selected by the user, the template list shows the sum of the templates that were found in all folders specified in TEMPLATEFOLDER commands.

## Script-related Keywords

The keywords in this section are used to specify the location of scripts, set up custom menus, set script variables and specify scripts to be executed under certain conditions.

### Specifying a Source Folder for Scripts - SCRIPTFOLDER

```
SCRIPTFOLDER folderName;
```

This keyword is used to specify a folder to search for scripts. Items in this folder will be added to the list that is displayed when the Scripter item is selected in the Tools menu. *folderName* is in the format specified in "Specifying Folder Names in the Setup File" on page 388.

## Adding a Custom Script Menu - SCRIPTMENU

```
SCRIPTMENU command, menu, scriptFile;
```

This keyword allows you to add a custom menu command to the DesignWorks menu bar and associate a script with it. In format given above, *menu* will be replaced by the title of the menu that the command is to appear in, and *command* will be replaced with the desired text of the menu item itself. If a menu with the specified title already exists, the new item will be added to the bottom of the existing menu.

**NOTE: Some non-alphanumeric characters are not allowed in menu titles or menu items since they convey special status information for the Macintosh menu manager. These include opening and closing parentheses "()" and the dash "-".**

The *scriptFile* parameter specifies the name of the script file to execute. The root directory for this file specification is the directory containing the setup file itself. Thus, is a simple script file name is given, the file will be searched for in the same directory as the setup file. Normally, a relative path should be specified to a folder within this folder, as shown in the examples below.

When the user selects the menu item, the script is executed for the current design. It is the responsibility of the script to determine if there is even a current design and if it is of the appropriate type. Examples of the usage of this keyword can be found in many of the design kits supplied with the package, such as the following:

```
SCRIPTMENU Generate Netlist, Generic PCB, :Scripts:GPCB Netlist
Script;
SCRIPTMENU Error Check, Generic PCB, :Scripts:GPCB Error Script;
SCRIPTMENU Enter Package Code, Generic PCB, :Scripts:GPCB Pkg
Script;
```

***See "Using Design Kits" on page 336 for more information on design types and design kit organization.***

## Specifying Global Script Variables - SCRIPTVAR

```
SCRIPTVAR varName, text;
```

This keyword allows you to specify the name and value for a script variable that is to be initialized each time a script is run. *varName* is the name of the variable. This must a legal script variable name, i.e. no more than 16 characters and only alphabetic, numeric, "_", or "." characters.

**NOTE: No checking is done for overlap between the name given here and any name that may appear in any script the user might run. If a variable with the same name is assigned a value in a script, the value assigned in the script will override the one specified here for the duration of the script. Global variable values are re-initialized each time a script is run.**

The *text* value specified will be assigned to the given variable each time a script is started. The text value will consist of all text on this line between the comma and the terminating semicolon, with leading and trailing spaces removed.

Here is an example of the usage of this keyword:

```
SCRIPTVAR _GPCB_Dir, :Scripts:;
```

## Running a Script Each Time a Design is Opened - OPENSCRIPT

```
OPENSCRIPT scriptFile;
```

This keyword registers a script to be executed each time any design is opened. This can be used, for example, to prompt the user for revision information, check for up to date attributes, etc.

**WARNING: In the current implementation, only one such script can be registered. If OPENSCRIPT keywords appear in more than one setup file, the last one executed will be used. No warnings are given. It is the script's responsibility to determine what type of design it is operating on and whether the operations it will perform are appropriate.**

## Specifying a Source Folder for Error Scripts - ERRORSCRIPTFOLDER

```
ERRORSCRIPTFOLDER folderName;
```

This keyword is used to specify a folder to search for scripts used by the ErrorScript tool. Items in this folder will be added to the list that is displayed when the ErrorScript item is selected in the Tools menu. *folderName* is in the format specified in “Specifying Folder Names in the Setup File” on page 388.

# Specifying Libraries to Open at Startup

The keywords in this section allow you to specify libraries to open automatically each time DesignWorks is started.

## Single Libraries

```
LIBRARY libName;
```

This specifies the name of a library to open. The library will be opened each time DesignWorks is started up and its name will appear in the Parts palette. As discussed above, libName can be a simple file name or a complete pathname specifying the disk and folders containing the library. Note that the name specified in the last FOLDER statement is always attached to the front of the name specified in the LIBRARY statement, even if a complete pathname is given. Following are examples of acceptable LIBRARY statements (these all assume no FOLDER statement has been used):

```
LIBRARY LibDisk:Counters;
```

specifies a library called Counters on a disk called LibDisk.

```
LIBRARY Discretes;
```

specifies a library called Discretes in the folder DesignWorks is in (or in the folder specified by the last FOLDER statement.

## All Libraries in a Folder

```
LIBRARYFOLDER folderName;
```

The LIBRARYFOLDER keyword names a folder to be searched for libraries. All libraries in this folder will be opened. Folders inside this folder are not checked. *folderName* is in the format specified in "Specifying Folder Names in the Setup File" on page 388.

# Auto-backup Operations

```
AUTOBACKUP minutes;
AUTOCHECKPOINT minutes;
```

The AUTOBACKUP and AUTOCHECKPOINT keywords enable the automatic backup and checkpoint facilities. More information on these commands is provided in "Backup Procedures" on page 64.

# Miscellaneous Settings

The following sections describe a number of keywords for customizing various aspects of program operation.

## Color Settings

```
COLOURS colourList;
```

*or*

```
COLORS colorList;
```

(depending on whose dictionary you use)

This specifies colors to be used in displaying objects on the Macintosh II screen. The statement has no effect on other machines. Each number in the list corresponds to a different type of circuit object. The COLOUR keyword is followed by a number of integers which will determine the colour displayed on the screen for various types of objects. If less numbers then object types appear in the COLOUR statement, the remaining ones are left unchanged from the defaults, which are specified in a resource entry called CTAB in the DesignWorks file. Extra numbers on the line are ignored.

Valid colour numbers are determined by the Macintosh system, and must be from the following list:

BLACK
WHITE
RED
GREEN
BLUE
CYAN
MAGENTA
YELLOW

Due to a quirk in the Macintosh system, device pictures can only be displayed in the colour they were originally created in, which is usually black. The overhead required to overcome this problem would considerably slow down screen updates, so we have not implemented it.

These colors are read in the following order:

| | |
|---|---|
| 1 | Default colour for items not specified |
| 2 | Name on a selected signal |
| 3 | Name on an unselected signal |
| 4 | Pin numbers on selected signal |
| 5 | Pin numbers on unselected signal |
| 6 | Reserved |
| 7 | New internally-created device symbols |
| 8 | Pin line in selected signal |
| 9 | Pin line in unselected signal |
| 10 | Non-pin line in selected signal |
| 11 | Non-pin line in unselected signal |
| 12 | Delay in selected device |
| 13 | Delay in unselected device |
| 14 | Name in selected device |
| 15 | Name in unselected device |

| | |
|---|---|
| 16 | Drawing boundary |
| 17 | Drawing grid |
| 18 | Print page outlines |
| 19 | Random text |
| 20 | Frame on random text |
| 21 | Rulings on random text |
| 22 | Frame on random pictures |
| 23 | Reserved |
| 24 | Reserved |
| 25 | Reserved |
| 26 | Reserved |
| 27 | Reserved |
| 28 | Reserved |
| 29 | Reserved |
| 30 | Reserved |
| 31 | Reserved |
| 32 | Reserved |
| 33 | Reserved |
| 34 | Reserved |
| 35 | Reserved |
| 36 | Reserved |
| 37 | Reserved |
| 38 | Pin line in selected bus |
| 39 | Pin line in unselected bus |
| 40 | Non-pin line in selected bus |
| 41 | Non-pin line in unselected bus |

**IMPORTANT: The SCALES, NORMALSIZE, PINSPACE, and NOERRORCHECK items following are provided for experimental use by experienced users of DesignWorks. The program is optimized for the default settings and is not guaranteed to adjust correctly to all possible changes. Test out the system thoroughly with any new settings before**

**performing any important tasks.**

## Setting Schematic Zoom Factors

### Reduce/Enlarge Settings

```
SCALES n1..n11;
```

The SCALES keyword is used to specify the magnification levels used by the Reduce and Enlarge commands. The keyword is followed by 1 to 11 decimal integers separated by blanks and sorted in ascending order. The 1:1 scale level (at which externally created pictures appear in their original size) is 14. Enlargements are specified by smaller numbers (e.g. 7 gives 200%) and reductions by larger numbers. Since all scaling is done using integer arithmetic, the best results are obtained at scale factors which are multiples or factors of 14. The default values are:

```
SCALES 4 7 10 14 18 24 28 42 63 98 140;
```

### Normal Size Setting

```
NORMALSCALE n;
```

The NORMALSCALE keyword is used to specify which of the scale steps specified in the SCALES line will be used as the "Normal Size" setting. The keyword must be followed by a single decimal integer in the range 1 to 11 which is the ordinal number of the scale step to be used as normal. The default is 4.

## Pin Spacing

```
PINSPACE n;
```

The PINSPACE keyword is used to specify the spacing between adjacent pins when breakout symbols are created. This can also serve as the default for symbols created by other tools. The keyword must be followed by a single decimal integer which is a multiple of the standard grid space of 5 dots. The default value is 2.

## Setting Device Placement Drawing Method

```
FLICKERDRAG;
```

This keyword causes the changes the way device symbols or pasted circuit segments are drawn while being placed. Without this keyword, the item is drawn in position and remains static until moved. With the FLICKERDRAG keyword placed in the setup file, the draw mode reverts to the flickering draw/undraw method used in pre-4.0 versions.

## Disabling Date Stamping

```
NODATESTAMP;
```

This keyword disables the automatic date stamping of devices that is done during editing operations.

***For more information on date stamping, see "Device Date Stamping" on page 153.***

## Disabling Loose End Markers

```
NOLOOSEENDS;
```

This keyword disables the automatic generation of "loose end" markers when signal lines are drawn in a schematic. These are the small "T" bars drawn on any signal line segment that does not touch any other segment.

## Internal Error Checking

```
NOERRORCHECK;
```

This option disables the internal error checking described Appendix Appendix H—Technical Support on page 401.

**NOTE: This should only be done in consultation with our Technical Support department as it will prevent warnings from being issued for internal or unusual system errors. In many cases these errors can be easily recovered if corrected immediately but may cause data corruption if left undetected.**

# Windows Font Translations - WINFONT

```
WINFONT winFontName, winFontSize, macFontName, macFontSize;
```

The WINFONT keyword allows you to specify font translations to make when transferring design files created by DesignWorks for Windows to the Macintosh version. The parameters of this command are:

| | |
|---|---|
| **winFontName** | The name of the font as used in the Windows system. The name must include any spaces or punctuation existing in the Windows font name, although letter case is not important. |
| **winFontSize** | The size of font to translate. If this is omitted (i.e. two commas) or set to zero, then any font with this name will be translated. |
| **macFontName** | This is the name of the Macintosh font to translate to. |
| **macFontSize** | This is the font size to use on the Macintosh. If this is omitted or set to zero, the size specified in the Windows file will be used. |

# Appendix E—Technical Support

Our goal is to ensure that you get reliable and productive use out of your DesignWorks package. If you have any problems or questions, please contact us at:

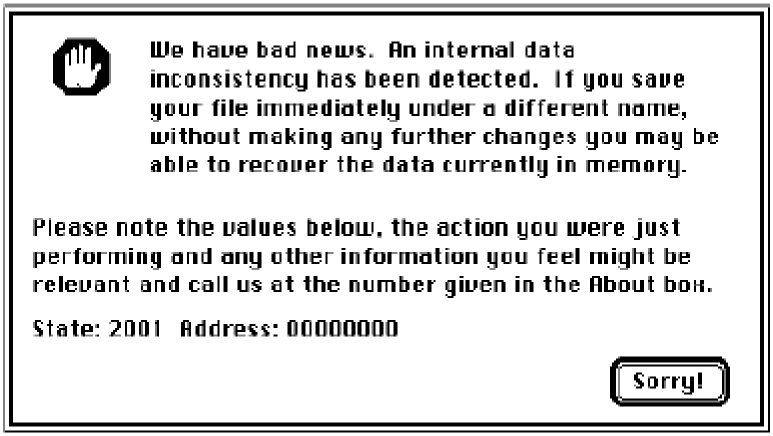
Technical Support,
Capilano Computing,
2631 Viking Way, Unit 218,
Richmond, B. C.
Canada, V3L 5G2

Voice (604) 522-6200
email tech@capilano.com
WWW http://www.capilano.com

## Internal Error Detection

DesignWorks makes use of a number of complex internal data structures in order to maintain an up-to-date image of your circuit at all times. To assist in detecting problems due to hardware failures, program errors or operating system errors, a code module has been added which checks these structures for consistency. This is done in the "background" while the program is idling and should normally be invisible to the user.

Should a problem be detected, a warning box similar to the following will be displayed:



We have bad news. An internal data inconsistency has been detected. If you save your file immediately under a different name, without making any further changes you may be able to recover the data currently in memory.

Please note the values below, the action you were just performing and any other information you feel might be relevant and call us at the number given in the About box.

State: 2001 Address: 00000000

The "State" value is a code that specifies the type of problem detected, and the "Address" value is the memory address of the object that was in error. It is beyond the scope of this manual to discuss the meaning of all possible error codes and an estimate of their severity. In general, a State < 100 indicates a structural problem that is likely to cause a serious program malfunction if you proceed with editing. The warning box will only appear once if this type of error is detected, even if other errors occur later. The error detection mechanism is reset when all design files are closed. States > 100 indicate unexpected situations detected in connection with some specific function and may or may not be serious. If the error disappears after a later check (due to an offending object being deleted, for example) you will be notified. This may occur if you delete the corrupted part of the circuit or if some other internal check succeeds in correcting the problem.

If you see this box in the course of normal program operation, then save your design file immediately <u>under a different name</u> (so as not to wipe out your last good backup). Quitting the program and rereading the saved file may result in the problem being corrected. If you can isolate the problem down to one specific object, then try deleting that object and recreating it. In any case, please contact our Technical Support department and provide as much information as you can about the situation which created the problem. We will help you in any way we can to recover any lost data.

Error checking can be disabled using the NOERRORCHECK setup file option, described in “Internal Error Checking” on page 399. This should not be used under normal circumstances.

# Chapter 404

# Index

When there are many entries for a topic, the primary entry is shown in **bold** face.

Commands and keywords starting with a "$" or "&" symbol are listed alphabetically, ignoring the starting symbol.